

FILEID**DZDRIVER

C 12

DDDDDDDD DDDDDDDDD ZZZZZZZZZZ DDDDDDDDD RRRRRRRRR I IIIII VV VV EEEEEEEEEE RRRRRRRR
DDDDDDDD DDDDDDDDD ZZZZZZZZZZ DDDDDDDDD RRRRRRRRR I IIIII VV VV EEEEEEEEEE RRRRRRRR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DD DD ZZ DD DD RRRRRRRR I II VV VV EEEEEEEE RRRRRRRR
DD DD ZZ DD DD RRRRRRRR I II VV VV EEEEEEEE RRRRRRRR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DD DD ZZ DD DD RR RR I II VV VV EE RR RR
DDDDDDDD ZZZZZZZZZZ DDDDDDDDD RRRRRRRRR I IIIII VV VV EEEEEEEEEE RR RR
DDDDDDDD ZZZZZZZZZZ DDDDDDDDD RRRRRRRRR I IIIII VV VV EEEEEEEEEE RR RR

(1)	136	DECLARATIONS
(1)	287	REGISTER DEFINITIONS
(1)	346	CONTROLLER INITIALIZATION
(1)	427	UNIT INITIALIZATION
(1)	512	MAINTENANCE ROUTINES
(1)	577	OUTPUT MODEM CONTROL
(1)	631	DZ-11 MODEM POLLER
(1)	717	RECEIVER INTERRUPT SERIVCE
(1)	891	START I/O ROUTINE
(1)	943	PORT ROUTINES STOP,RESUME,XON,XOFF
(1)	1085	OUTPUT INTERRUPT SERVICE
(1)	1348	SET SPEED, PARITY PARAMETERS
(1)	1400	INITIALIZE DZ-11 MODEM POLLING

```
0000 1 .if df DZV
0000 2 .iff   .TITLE DZVDRIVER - Port Driver for DZV-11 support
0000 3 .iff   .TITLE DZDRIVER - Port Driver for DZ-11 support
0000 4 .endc
0000 5 .IDENT 'V04-000'
0000 6
0000 7
0000 8 :
0000 9 ****
0000 10 *
0000 11 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 12 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 13 * ALL RIGHTS RESERVED.
0000 14 *
0000 15 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 16 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 17 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 18 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 19 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 20 * TRANSFERRED.
0000 21 *
0000 22 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 23 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 24 * CORPORATION.
0000 25 *
0000 26 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 27 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 28 *
0000 29 *
0000 30 ****
0000 31
0000 32 ++
0000 33 FACILITY:
0000 34
0000 35     VAX/VMS TERMINAL DRIVER
0000 36
0000 37 ABSTRACT:
0000 38
0000 39     DZ PORT DRIVER
0000 40     This module functions as a port driver for DZ11 and DZ32 terminal
0000 41     controllers. It contains hardware specific port level service
0000 42     routines.
0000 43
0000 44 AUTHOR:
0000 45
0000 46     RICK SPITZ
0000 47
0000 48 Revision history:
0000 49
0000 50     V03-028 MIR0480 Michael I.Rosenblum 8-Aug-1984
0000 51     Fix bugs found in testing the DZ-32 and reported in
0000 52     QAR 1126 (FT1).
0000 53
0000 54     V03-027 LMP0275 L. Mark Pilant, 12-Jul-1984 21:01
0000 55     Initialize the ACL info in the ORB to be a null descriptor
0000 56     list rather than an empty queue. This avoids the overhead
0000 57     of locking and unlocking the ACL mutex, only to find out
```

0000 58 : that the ACL was empty.
0000 59 :
0000 60 : V03-026 EMD0090 Ellen M. Dusseault 30-Apr-1984
0000 61 : Add DEV\$M_NNM characteristic to DEVCCHAR2 so that these
0000 62 : devices will have the "node\$" prefix.
0000 63 :
0000 64 : V03-025 LMP0221 L. Mark Pilant, 7-Apr-1984 13:37
0000 65 : Change UCBSL_OWNNUIC to ORBSL_OWNER and UCBSW_VPROT to
0000 66 : ORBSW_PROT.
0000 67 :
0000 68 : V03-024 JLV0321 Jake VanNoy 5-JAN-1984
0000 69 : Minor enhancements to DZV11 support. Comment out
0000 70 : DPT_STORE for parity, must fix a restriction.
0000 71 :
0000 72 : V03-023 MIR0055 Michael I. Rosenblum 30-June-1983
0000 73 : Remove code from unit and controller-init routines
0000 74 : and make insert calls to class driver macros
0000 75 : Add DZV11 support.
0000 76 :
0000 77 : V03-022 RKS0022 RICK SPITZ 14-MAR-1983
0000 78 : ADD ENHANCEMENTS TO SUPPORT LOGICAL UCB.
0000 79 :
0000 80 : V03-021 MIR0022 Michael I. Rosenblum 19-Jan-1982
0000 81 : Change references to UCBSB_ERTCNT to use UCBSW_TT_UNITBIT
0000 82 : to be more maintainable.
0000 83 : Replace old vector table with new vector table macro.
0000 84 : Remove references to UCBSL_DEVDEPEND and UCBSQ_TT_STATE
0000 85 : move these references into the class driver jacket routines
0000 86 :
0000 87 : V03-020 MIR0021 Michael I. Rosenblum 17-Jan-1983
0000 88 : Fix DZ32 DZ_SET table entry.
0000 89 :
0000 90 : V03-019 RKS0019 RICK SPITZ 13-JAN-1983
0000 91 : Repair problem with port vector macro
0000 92 :
0000 93 : V03-018 MIR0019 Michael I. Rosenblum 11-Jan-1982
0000 94 : Fix undefined symbol created by MIR0018.
0000 95 :
0000 96 : V03-017 MIR0018 Michael I. Rosenblum 07-Jan-1983
0000 97 : Change the port vector table to use the SVEC macros
0000 98 :
0000 99 : V03-016 MIR0017 Michael I. Rosenblum 05-Jan-1983
0000 100 : Add powerfail check in the Unit init routine to allow the
0000 101 : terminal class driver to take positive action on powerfail.
0000 102 : Change code to accept a byte value as returns from the
0000 103 : GETNXT and PUTNXT class services, this will removes this
0000 104 : information from the condition codes.
0000 105 :
0000 106 : V03-015 MIR0016 Michael I. Rosenblum 29-Dec-1982
0000 107 : Replace time calculation code with TIMSET macro call
0000 108 :
0000 109 : V03-014 MIR0015 Michael I. Rosenblum 20-Dec-1982
0000 110 : Remove entry point to reflect the redefinition of the PORT_DISCONNECT
0000 111 : entry point.
0000 112 :
0000 113 : V03-013 MIR0014 Michael I. Rosenblum 17-Dec-1982
0000 114 : Remove code to calculate flow control characters from

0000 115 : port XON and XOFF routines and move that code into
0000 116 : the class driver.
0000 117 :
0000 118 : V03-012 RKS0012 RICK SPITZ 16-SEP-1982
0000 119 : Check reference count in unit init to determine if
0000 120 : modem control should be initialized or hungup. This
0000 121 : is needed to insure that a hangup ^Y is posted on
0000 122 : powerfail for modem lines.
0000 123 :
0000 124 : V03-011 JLV0211 Jake VanNoy 2-JUL-1982
0000 125 : Remove check for powerfail in unit init that prevents
0000 126 : SETUP UCB from being called. This insures that UCB fields
0000 127 : are initialized correctly when Unit init is called for
0000 128 : use with CSS unibus switch.
0000 129 :
0000 130 : V03-010 KDM0002 Kathleen D. Morse 28-Jun-1982
0000 131 : Added \$DEVDEF, \$IPLDEF, \$PRDEF, and \$SSDEF.
0000 132 :
0000 133 :
0000 134 ;--

```

0000 136 .SBTTL DECLARATIONS
0000 137
0000 138
0000 139 ; EXTERNAL DEFINITIONS:
0000 140 ;
0000 141 $ACBDEF : DEFINE ACB
0000 142 $CRBDEF : DEFINE CRB
0000 143 $DCDEF : DEVICE DEFINITIONS
0000 144 $DDDBDEF : DEFINE DDB
0000 145 $DEVDEF : DEFINE DEVICE TYPES
0000 146 $DYNDEF : DYNAMIC STRUCTURE DEFINITONS
0000 147 $IDBDEF : DEFINE IDB OFFSETS
0000 148 $IODEF : DEFINE I/O FUNCTION CODES
0000 149 $IPLDEF : DEFINE INTERRUPT PRIORITY LEVELS
0000 150 $IRPDEF : IRP DEFINITIONS
0000 151 $ORBDEF : DEFINE OBJECT'S RIGHTS BLOCK OFFSETS
0000 152 $PRDEF : DEFINE PROCESSOR REGISTERS
0000 153 $SSDEF : DEFINE SYSTEM STATUS CODES
0000 154 $TTYDEF : DEFINE TERMINAL DRIVER SYMBOLS
0000 155 $TTDEF : DEFINE TERMINAL TYPES
0000 156 $TT2DEF : DEFINE EXTENDED DEFINITIONS
0000 157 $TQEDEF : DEFINE TIMER QUEUE OFFSETS
0000 158 $UCBDEF : DEFINE UCB
0000 159 $UBADEF : DEFINE UBA
0000 160 $VECDEF : DEFINE VECTOR FOR CRB
0000 161 $TTYMACS : DEFINE TERMINAL DRIVER MACROS
0000 162 $TTYDEFS : DEFINE TERMINAL DRIVER SYMBOLS
0000 163 $TTYMODEM : DEFINE MODEM DEFINITIONS
0000 164
0000 165
0000 166 ; LOCAL STORAGE
0000 167
0000 168 .PSECT $$S105_PROLOGUE
0000 169
0000 170
0000 171 ; Driver prologue table:
0000 172
0000 173 ;
0000 174
0000 175 DZ$DPT:; ; DRIVER START
0000 176 .IF DF DZV
0000 177 DPTAB - ; DRIVER PROLOGUE TABLE
0000 178 END=DZ$END,- ; End and offset to INIT's vectors
0000 179 UCBSIZE=UCB$C TT LENGTH,- ; SIZE OF UCB
0000 180 FLAGS=DPTSM_NOUNLOAD,- ; DO NOT ALLOW UNLOAD
0000 181 ADAPTER=UBA,- ; ADAPTER TYPE
0000 182 DEFUNITS=4,- ; DZV has 4 units
0000 183 NAME=DZDRIVER,- ; NAME OF DRIVER
0000 184 VECTOR=PORT_VECTOR ; PORT DRIVER VECTOR TABLE
0000 185 .IFF
0000 186 DPTAB - ; DRIVER PROLOGUE TABLE
0000 187 END=DZ$END,- ; End and offset to INIT's vectors
0000 188 UCBSIZE=UCB$C TT LENGTH,- ; SIZE OF UCB
0000 189 FLAGS=DPTSM_NOUNLOAD,- ; DO NOT ALLOW UNLOAD
0000 190 ADAPTER=UBA,- ; ADAPTER TYPE
0000 191 DEFUNITS=8,- ; Number of units to create
0000 192 NAME=DZDRIVER,- ; NAME OF DRIVER

```

```

0000 193 VECTOR=PORT_VECTOR : PORT DRIVER VECTOR TABLE
0038 194 .ENDC
0038 195 DPT_STORE INIT
0038 196 DPT_STORE UCB,UCBSB_FIPL,B,8 : FORK IPL
003C 197 DPT_STORE UCB,UCBSL_DEVCHAR,L,<-: CHARACTERISTICS
003C 198 DEVSM_REC|-|
003C 199 DEVSM_AVL|-|
003C 200 DEVSM_IDV|-|
003C 201 DEVSM_ODV|-|
003C 202 DEVSM_TRM|-|
003C 203 DEVSM_CCL>
0043 204 DPT_STORE UCB,UCBSL_DEVCHAR2,L,- : Device Characteristics
0043 205 <DEVSM_NNM> : prefix with "nodes"
004A 206 DPT_STORE UCB,UCBSB_DEVCLASS,B,DC$ TERM;
004E 207 DPT_STORE UCB,UCBSB_TT_DETYPE,B,TT$ UNKNOWN : TYPE
0052 208 DPT_STORE UCB,UCBSW_TT_DESIZE,B,TT$UNKNOWN : BUFFER SIZE
0059 209 DPT_STORE UCB,UCBSL_TT_DECHAR,BL,TTYSGL_DEFCHAR : DEFAULT CHARACTERS
0060 210 DPT_STORE UCB,UCBSL_TT_DECHAI,BL,TTYSGL_DEFCHAR2 : DEFAULT CHARACTERS
0067 211 DPT_STORE UCB,UCBSW_TT_DESPEE,B,TTYSGB_DEF SPEED : DEFAULT SPEED
006E 212 DPT_STORE UCB,UCBSW_TT_DESPEE+1,B,TTYSGB_RSPEED : DEFAULT SPEED
0075 213 DPT_STORE UCB,UCBSB_TT_DEPARI,B,TTYSGB_PARITY : DEFAULT PARITY
007C 214 DPT_STORE UCB,UCBSB_TT_PARITY,B,TTYSGB_PARITY : DEFAULT PARITY
007C 215 DPT_STORE UCB,UCBSB_DEVTYPE,B,TT$ UNKNOWN : TYPE
0080 216 DPT_STORE UCB,UCBSW_DEVBUFSIZ,B,TTYSGW_DEFBUF : BUFFER SIZE
0087 217 DPT_STORE UCB,UCBSL_DEVDEPEND,BL,TTYSGL_DEFCHAR : DEFAULT CHARACTERS
008E 218 DPT_STORE UCB,UCBSL_DEVDEPND2,BL,TTYSGL_DEFCHAR2 : DEFAULT CHARACTERS
0095 219 DPT_STORE UCB,UCBSW_TT_SPEED,B,TTYSGB_DEF SPEED : DEFAULT SPEED
009C 220 DPT_STORE UCB,UCBSW_TT_SPEED+1,B,TTYSGB_RSPEED : DEFAULT SPEED
00A3 221 DPT_STORE UCB,UCBSB_DIPL,B,21 : DEVICE IPL
00A7 222 DPT_STORE UCB,UCBSL_TT_WFLINK,L,0 : Zero write queue.
00AE 223 DPT_STORE UCB,UCBSL_TT_WBLINK,L,0 : Zero write queue.
00B5 224 DPT_STORE UCB,UCBSL_TT_RTIMOU,L,0 : Zero read timed out disp.
00BC 225 DPT_STORE ORB,ORB$B_FLAGS,B,- : Protection block flags
00BC 226 >ORB$M PROT 16> : SOGW protection word
00C0 227 DPT_STORE ORB,ORB$W_PROT,B,TTYSGW_PROT : Default allocation protection
00C7 228 DPT_STORE ORB,ORB$L_OWNER,BL,TTYSGC_OWNNUIC : Default owner UIC
00CE 229 DPT_STORE DDB,DDB$L_DDT,D,DZ$DDT
00D3 230
00D3 231 DPT_STORE REINIT
00D3 232 DPT_STORE CRB,CRBSL_INTD+VECSL_INITIAL,D,DZ$INITIAL : CONTROLLER INIT
00D8 233 DPT_STORE CRB,CRBSL_INTD+VECSL_UNITINIT,D,DZ$INITLINE: UNIT INIT
00DD 234 DPT_STORE END
0000 235
0000 236 DDTAB DEVNAM = DZ,- ; DUMMY DZ PORT DRIVER DISPATCH TABLE
0000 237 START = 0,-
0000 238 FUNCTB = 0
0038 239 .PSECT $SS$115_DRIVER
0038 240
0038 241
0038 242 : THE ASSOCIATED CLASS DRIVER USES THIS TABLE TO COMMAND THE PORT DRIVER.
0038 243 : THE ADDRESS OF THIS TABLE IS CONTAINED IN THE TERMINAL UCB EXTENSION AREA.
0038 244 : THE OFFSET DEFINITIONS ARE DEFINED BY TTYDEFS.
0038 245 :
0038 246 :
0038 247 :
0038 248 PORT_VECTOR:
0038 249

```

0038 250 :
0038 251 : DZ-11 SPECIFIC DISPATCH TABLE
0038 252 :
0038 253 \$VECINI DZ11,DZ\$NULL
0070 254 \$VEC STARTIO,DZ11\$STARTIO
003C 255 \$VEC SET LINE,DZ\$SET LINE
0044 256 \$VEC DS SET,DZ11\$DS_SET
0048 257 \$VEC XON,DZ11\$XON
004C 258 \$VEC XOFF,DZ11\$XOFF
0050 259 \$VEC STOP,DZ\$STOP
0054 260 \$VEC ABORT,DZ\$ABORT
005C 261 \$VEC RESUME,DZ11\$RESUME
0060 262 \$VEC SET MODEM,DZ11\$SET_MODEM
0064 263 \$VEC MAINT,DZ11\$MAINT
006C 264 .IF NDF DZV
006C 265 SVECEND END=NO
0070 266 :
0070 267 : DZ-32 SPECIFIC DISPATCH TABLE
0070 268 :
0070 269 :
0070 270 \$VECINI DZ32,DZ\$NULL
00A8 271 \$VEC STARTIO,DZ32\$STARTIO : START NEW OUTPUT
0074 272 \$VEC SET LINE,DZ\$SET LINE : SET NEW PARITY/SPEED
007C 273 \$VEC DS SET,DZ32\$DS_SET : SET NEW OUTPUT MODEM SIGNALS
0080 274 \$VEC XON,DZ32\$XON : SEND XON
0084 275 \$VEC XOFF,DZ32\$XOFF : SEND XOFF
0088 276 \$VEC STOP,DZ\$STOP : STOP CURRENT OUTPUT
008C 277 \$VEC ABORT,DZ\$ABORT : ABORT CURRENT OUTPUT
0094 278 \$VEC RESUME,DZ32\$RESUME : RESUME STOPPED OUTPUT
0098 279 \$VEC MAINT,DZ32\$MAINT : INVOKE MAINTENANCE FUNCTIONS
00A4 280 .ENDC
00A4 281 SVECEND
00AC 282 DZ\$NULL: : NULL PORT ROUTINE
00AC 283 RSB
00AD 284
00AD 285

05

00AD 287 .SBTTL REGISTER DEFINITIONS
00AD 288
00AD 289
00AD 290 : CSR BIT DEFINITIONS (CSR) (READ/WRITE)
00AD 291 :
00AD 292 \$VIELD DZCSR,0,<-
00AD 293 <MODE,1,M>,- : DZ32 - MODE/ DZ11 - UNUSED
00AD 294 <DS_ENAB,1,M>,- : DZ32 - DATA SET INTERRUPT ENABLE
00AD 295 <,1> - : UNUSED
00AD 296 <MAINF,1,M>,- : LINE TURNAROUND
00AD 297 <CLEAR,1,M>,- : MASTER RESET
00AD 298 <MASTENAB,1,M>,- : MASTER SCAN ENABLE
00AD 299 <RCVINT,1,M>,- : RECEIVER INTERRUPT ENABLE
00AD 300 <RCVRDY,1,M>,- : RECEIVER READY
00AD 301 <LINE,3,M>,- : LINE NUMBAE (0 - 7)
00AD 302 <DS_CHG,1,M>,- : DZ32 - DATA SET INTERRUPT
00AD 303 <,2> - : UNUSED
00AD 304 <SNDINT,1,M>,- : TRANSMIT INTERRUPT ENABLE
00AD 305 <SNDRDY,1,M>,- : TRANSMITTER READY
00AD 306 >
00AD 307 :
00AD 308 : RECEIVER BUFFER (CSR+2) (READ ONLY)
00AD 309 :
00AD 310 \$VIELD DZRCV,0,<-
00AD 311 <BUF,8,M>,- : RECEIVER DATA
00AD 312 <LINE,3,M>,- : LINE NUMBER (0 - 7)
00AD 313 <,1> -
00AD 314 <PARERR,1,M>,- : PARITY ERROR
00AD 315 <FRAMER,1,M>,- : FRAME ERROR
00AD 316 <OVERRUN,1,M>,- : OVERRUN ERROR
00AD 317 <VALID,1,M>,- : DATA VALID
00AD 318 >
00AD 319 :
00AD 320 : LINE PARAMETER REGISTER (CSR+2) (WRITE ONLY)
00AD 321 :
00AD 322 :
00AD 323 \$VIELD DZLPR,0,<-
00AD 324 <LINE,3,M>,- : LINE NUMBER (0-7)
00AD 325 <SIZE,2,M>,- : CHARACTER SIZE
00AD 326 <STOP,1,M>,- : NUMBER STOP BITS
00AD 327 <PARITY,1,M>,- : PARITY ENABLE
00AD 328 <ODD,1,M>,- : ODD PARITY
00AD 329 <SPEED,4,M>,- : LINE SPEED
00AD 330 <CLOCK,1,M>,- : RECEIVER CLOCK
00AD 331 <SPLIT,1,M>,- : DZ32 - SPLIT SPEED
00AD 332 >
00AD 333 :
00AD 334 : DZ-32 SPECIFIC MODEM CONTROL
00AD 335 :
00AD 336 :
00AD 337 :
00AD 338 \$VIELD DZLCS1,8,<-
00AD 339 <,7> -
00AD 340 <ACK,1,M>,- : READY FOR COMMAND/ UPDATE OUTPUT MODEM
00AD 341 >
00AD 342
00AD 343

DZDRIVER
V04-000

- Port Driver for DZ-11 support
REGISTER DEFINITIONS

L 12

OODAD 344

16-SEP-1984 02:24:50 VAX/VMS Macro V04-00
5-SEP-1984 04:15:55 [TTDRV.R.SRC]DZDRIVER.MAR;1

Page 8
(1)

DZ
VO

00AD 346 .SBTTL CONTROLLER INITIALIZATION
 00AD 347
 00AD 348
 00AD 349
 00AD 350 ++
 00AD 351 : DZSINITIAL - INITIALIZE INTERFACE
 00AD 352
 00AD 353
 00AD 354
 00AD 355 : FUNCTIONAL DESCRIPTION:
 00AD 356 : THIS ROUTINE IS ENTERED AT SYSTEM STARTUP AND POWER RECOVERY.
 00AD 357 : INPUTS:
 00AD 358 :
 00AD 359 : R4 = ADDRESS OF THE UNIT CSR
 00AD 360 : R5 = IDB OF UNIT
 00AD 361 : R8 = ADDRESS OF THE UNIT CRB
 00AD 362 :
 00AD 363 : OUTPUTS:
 00AD 364 :
 00AD 365 : R2 is destroyed.
 00AD 366 :
 00AD 367 : IMPLICIT INPUTS:
 00AD 368 :
 00AD 369 : IPL = IPL\$POWER
 00AD 370 :
 00AD 371 --
 00AD 372 DZSINITIAL:: : INITIALIZE DZ UNIT
 00AD 373 :
 00AD 374 : SET UP CONTROLLER
 00AD 375 :
 00AD 376 :
 00AD 377 : class_ctrl_init dz\$dpt,port_vector
 00DA 378 :
 64 10 B0 00DA 379 25\$: MOVW #DZCSRSM_CLEAR,(R4) : INIT CONTROLLER RESET
 00DD 380 :
 00DD 381 :
 00DD 382 : WAIT TILL CONTROLLER INITIALIZATION IS COMPLETE
 00DD 383 :
 00DD 384 : TIMEWAIT #500,#DZCSRSM_CLEAR,(R4),W,,FALSE.
 0104 385 :
 64 4063 8F B0 0104 386 :
 0104 387 :
 0109 388 :
 0109 389 :
 0109 390 :
 3E 50 E9 0109 391 :
 010C 392 :
 19 52 64 B0 010C 393 :
 19 52 00 E0 010F 394 :
 0112 395 :
 0B A8 42 8F 90 0112 396 : 100\$:
 0112 397 :
 0112 398 :
 0112 399 : INIT DZ-11 INTERRUPT VECTORS
 0112 400 : THIS IS DONE HERE TO ALLOW THE DRIVER TO SERVICE INTERRUPTS
 0112 401 : FOR BOTH THE DZ-11 AND DZ-32 BETWEEN CONTROLER AN UNIT INIT.
 0112 402 :

```

28 A8 000003E9'EF DE 0118 403 MOVAL DZ11$INTINP,CRBSL_INTD+4(R8) ; INIT RECEIVER VECTOR
4C A8 000006E9'EF DE 0120 404 MOVAL DZ11$INTOUT,CRBSL_INTD2+4(R8) ; INIT TRANSMITTER VECTOR
1C A8 B4 0128 405 CLRW CRBSB_DZ_RING(R8) ; RESET CURRENT DZ-11 MODEM STATE
      05 012B 406 RSB
      012C 407
      012C 408
      012C 409 110$: ; DZ-32 CONTROLLER SPECIFIC INIT
      012C 410 .if ndf DZV
      012C 411
0B A8 43 8F 90 012C 412 MOVB #DT$ DZ32,CRBSB_TT_TYPE(R8) ; CONTROLLER IS DZ-32
0E A5 94 0131 413 CLRB IDBSB_TT_ENABLETRST ; RESET DZ-32 LINE ENABLE
07 A4 0E A5 90 0134 414 MOVB IDBSB_TT_ENABLE(R5),7(R4) ; RESET TRANSMIT LINE ENABLES
      0139 415 :
      0139 416 ; INIT DZ-32 ALTERNATE INTERRUPT VECTORS
      0139 417 :
28 A8 00000477'EF DE 0139 418 MOVAL DZ32$INTINP,CRBSL_INTD+4(R8) ; INIT RECEIVER VECTOR
4C A8 000007A3'EF DE 0141 419 MOVAL DZ32$INTOUT,CRBSL_INTD2+4(R8) ; INIT TRANSMITTER VECTOR
      0149 420 .endc
      0149 421 RSB
      014A 422
      014A 423 DZSCTRL_ERROR:
      05 014A 424 RSB
      014B 425

```

014B 427 .SBTTL UNIT INITIALIZATION
 014B 428 ++ DZ\$INITLINE - UNIT INITIALIZATION
 014B 429
 014B 430
 014B 431
 014B 432
 014B 433
 014B 434
 014B 435
 014B 436
 014B 437
 014B 438
 014B 439
 014B 440
 014B 441
 014B 442
 014B 443
 014B 444 DZ\$INITLINE::
 014B 445
 54 24 A5 D0 014B 446 MOVL UCB\$L_CRB(R5),R4 : GET CRB ADDRESS
 014F 447
 014F 448 .IF NDF DZV : IF NOT DZV
 50 FF1D CF DE 014F 449
 0B A4 43 8F 91 0154 450 MOVAL DZ32\$VEC, R0 : SET DZ-32 PORT VECTOR TABLE
 05 13 0159 451 CMPB #DT\$_DZ32,CRB\$B_TT_TYPE(R4) ; IS IT DZ-32 ?
 015B 452 BEQL 58 ; YES
 015B 453
 015B 454 .ENDC : END OF DZ32 CODE
 015B 455
 50 FED9 CF DE 015B 456 MOVAL DZ11\$VEC, R0 : SET DZ-11 PORT VECTOR TABLE
 0160 457 5\$: CLASS_UNIT INIT
 53 64 A5 10 A8 01A9 458 BISW #UCBSM_ONLINE_UCBSW_STS(R5); SET ONLINE
 01 54 A5 78 01AD 459 10\$: ASHL UCBSW_ONIT(R5),#1,R3 : BUILD UNIT'S BIT MASK
 0106 C5 53 B0 01B2 460 MOVW R3,UCBSW_TT_UNITBIT(R5) : SAVE IT
 51 0114 C5 D0 01B7 461 MOVL UCB\$L_TT_CLASS(R5),R1 : ADDRESS CLASS VECTOR TABLE
 08 B1 16 01BC 462 JSB @CLASS_SETUP_UCB(R1) : INIT UCB FIELDS
 00000864'EF 16 01BF 463 20\$: JSB DZ\$SET_LINE : INIT SPEED/PARITY
 54 24 A5 D0 01C5 464
 01C9 465 MOVL UCB\$L_CRB(R5),R4 : GET CRB ADDRESS
 01C9 466
 01C9 467
 01C9 468 .IF NDF DZV : IF NOT DZV
 0B A4 42 8F 91 01C9 469
 69 13 01CE 470 CMPB #DT\$_DZ11,CRB\$B_TT_TYPE(R4) : CONTROLLER DZ11?
 01D0 471 BEQL 258 ; YES
 01D0 472
 01D0 473
 01D0 474
 01D0 475 : INIT RECEIVER MODEM STATUS FOR DZ-32
 01D0 476
 54 2C B4 D0 01D0 477 MOVL @CRB\$L_INTD+VEC\$L_IDB(R4),R4 : GET CSR ADDRESS
 01D4 478
 01D4 479 : WAIT TILL MODEM CONTROL READY FOR COMMND
 01D4 480
 01D4 481
 04 A4 56 50 E9 01FE 482
 54 A5 B0 0201 483
 TIMEWAIT #500,#DZLCS1SM_ACK,4(R4),W,,TRUE.
 BLBC R0,DZ\$UNIT_ERROR
 MOVW UCB\$W_UNIT(R5),4(R4) : REQUEST STATUS ON LINE

DZ
VO

0124 C5 24 50 E9 90	0206 484	
	0206 485 ; WAIT FOR COMPLETION	
	0206 486	
	0206 487	TIMEWAIT #500 #DZLCS1SM ACK,4(R4),W.,TRUE.
	0230 488	BLBC R0,DZ\$UNIT_ERROR
	0233 489	MOVBL 4(R4),UCBSB_TT_DS_RCV(R5) ; UPDATE RECEIVER MODEM STATUS
	0239 490	
	0239 491 25\$:	
	0239 492 .ENDC ; END OF DZ32 CODE	
	0239 493	
SU 0114 51 52 DD 00 9A	0238 494	PUSHL R2
OC 0C B0 16 52 8ED0	023F 495	MOVZBL #MODEMSC_INIT,R1 ; ASSUME INIT MODEM PROTOCOL
	0243 496	MOVL UCBSL_TT_CLASS(R5),R0 ; ADDRESS CLASS VECTOR TABLE
	0246 497	JSB ACCLASS_DS_TRAN(R0) ; INVOKE TO INIT MODEM PROTOCOL
08 64 A5 05 E1 0249 500	0249 498	POPL R2
50 0114 C5 00 00 00 00 00	024E 501	BBC #UCBSV_POWER,UCBSW_STS(R5),40\$; DID WE DETECT A POWER FAIL
20 B0 17 05 0253 0256 502	0253 502	MOVL UCBSL_TT_CLASS(R5)-R0 ; GET THE CLASS VECTOR TABLE ADDRESS
	0253 503 40\$: RSB	JMP ACCLASS_POWERFAIL(R0) ; AND GOTO THE POWERFAIL CODE
	0257 504	
	0257 505 ; ERROR DETECTED DURING INITIALIZATION	
	0257 506 :	
	0257 507	
64 A5 10 AA 0257 508	0257 508 DZ\$UNIT_ERROR:	
05 0258 509	0258 509 BICW #UCBSM_ONLINE,UCBSW_STS(R5) ; UNIT NOT ON LINE	
	0258 510 RSB	

025C 512 .SBTTL MAINTENANCE ROUTINES
 025C 513 ++ DZ\$MAINT - MAINTENANCE FUNCTIONS
 025C 514
 025C 515
 025C 516 FUNCTIONAL DESCRIPTION:
 025C 517
 025C 518 THIS ROUTINE PERFORMS MAINTENANCE FUNCTIONS FOR THE DZ .
 025C 519
 025C 520
 025C 521 INPUTS:
 025C 522
 025C 523 R5 = UBC ADDRESS
 025C 524 UCB\$B_TT_MAINT = FUNCTION TO BE PERFORMED
 025C 525
 025C 526 OUTPUTS:
 025C 527
 025C 528 --
 025C 529 IF NDF DZV
 025C 530 DZ32\$MAINT:
 012A 01 93 025C 531 BITB #IOSM_LOOPA-7 - ; LOOPBACK FUNCTION
 C5 025E 532 UCB\$B_TT_MAINT(R5)
 52 40 8F 06 13 0261 533 BEQL 5\$; NO
 OC 0263 534 MOVZBL #^X40,R2 ; SPECIFY LOOPBACK CODE
 11 0267 535 BRB 10\$
 012A 02 93 0269 536 5\$:
 C5 0268 537 BITB #IOSM_UNLOOPA-7,- ; RESET LOOPBACK FUNCTION
 OF 026E 538 UCB\$B_TT_MAINT(R5)
 52 7200 8F 3C 13 0270 539 BEQL 15\$; NO
 000002ED'EF 50 01 16 0275 540 MOVZWL #^X7200,R2 ; SPECIFY UNLOOP CODE (BOTH)
 9A 027B 541 10\$: JSB DZ32\$DS_SET ; UPDATE CONTROLLER
 05 027E 542 MOVZBL #1,R0 ; INDICATE SUCCESS
 RSB
 012A 20 93 027F 543 15\$:
 C5 0281 546 BITB #IOSM_LOOP_EXTA-7 - ; LOOPBACK FUNCTION
 06 0284 547 UCB\$B_TT_MAINT(R5)
 52 72 8F 9A 13 0286 548 BEQL 20\$; NO
 E9 11 028A 549 MOVZBL #^X72,R2 ; SPECIFY LOOPBACK CODE
 028C 550 BRB 10\$
 028C 551 20\$: ; CHECK OTHER FUNCTIONS
 028C 552 ENDC
 028C 553
 028C 554 DZ11\$MAINT:
 012A 04 93 028C 555 BITB #IOSM_LINE OFFA-7 - ; LINE OFF
 C5 028E 556 UCB\$B_TT_MAINT(R5)
 80 08 13 0291 557 BEQL 10\$; NO
 8F 88 0293 558 BISB #UCBSM TT_DSB_L - ; DISABLE LINE
 012A C5 0296 559 UCB\$B_TT_MAINT(R5)
 0D 11 0299 560 BRB 20\$
 012A 10 93 0298 561 10\$: ; LINE ON
 C5 029D 562 BITB #IOSM_LINE ONA-7 - ; LINE ON
 10 13 02A0 563 UCB\$B_TT_MAINT(R5)
 80 8F 8A 02A2 564 BEQL 30\$; NO
 012A C5 02A5 565 BICB #UCBSM TT_DSB_L - ; REENABLE LINE
 02A8 566 UCB\$B_TT_MAINT(R5)
 00000864'EF 16 02A8 567 20\$: JSB DZ\$SET_LINE ; IMPLEMENT FUNCTION
 568

DZDRIVER
V04-000

- Port Driver for DZ-11 support
MAINTENANCE ROUTINES

E 13

16-SEP-1984 02:24:50 VAX/VMS Macro V04-00
5-SEP-1984 04:15:55 [TTDRV.R.SRC]DZDRIVER.MAR;1

Page 14
(1)

50 01 9A 02AE 569
05 02B1 570
02B2 571 30\$: MOVZBL #1,RO
RSB
50 D4 02B2 572
05 02B4 573 CLRL R0
02B5 574
02B5 575 RSB

DZ
VO

.SBTTL OUTPUT MODEM CONTROL

DZSDS_SET - SET OUTPUT MODEM SIGNALS

FUNCTIONAL DESCRIPTION:

THIS ROUTINE OUTPUTS THE OUTPUT MODEM SIGNALS FOR THE SPECIFIED UNIT

INPUTS:

R2 = LOW BYTE - SIGNALS TO ACTIVATE
HIGH BYTE - SIGNALS TO DEACTIVATE

R5 = UBC ADDRESS

OUTPUTS:

R0-R3 ARE USED.

DZ11SDS_SET:

52	0125	C5	54	DD	02B5	597	PUSHL	R4	: SAVE
	52	F8	52	88	02B7	598	BISB	R2_UCBSB_TT_DS_TX(R5)	: SET NEW OUTPUT SIGNALS
	0125	C5	52	8A	02C1	599	ASHL	#-8,R2,R2	: ACCESS SIGNALS TO RESET
	54	24	A5	DO	02C6	600	BICB	R2_UCBSB_TT_DS_TX(R5)	: RESET THEM
	53	2C	B4	DO	02CA	601	MOVL	UCBSL_CRB(R5),R4	: GET CRB ADDRESS
						602	MOVL	ACRBSE_INTD+V ^E CSL_IDB(R4),R3	
51	0125	C5	01	01	02CE	603	--		
	51	51	54	A5	78	604	DZ11SDS_SET:		
1E	A4	0106	C5	8A	02D5	605	PUSHL	R4	: SAVE
						606	BISB	R2_UCBSB_TT_DS_TX(R5)	: SET NEW OUTPUT SIGNALS
						607	ASHL	#-8,R2,R2	: ACCESS SIGNALS TO RESET
						608	BICB	R2_UCBSB_TT_DS_TX(R5)	: RESET THEM
						609	MOVL	UCBSL_CRB(R5),R4	: GET CRB ADDRESS
						610	MOVL	ACRBSE_INTD+V ^E CSL_IDB(R4),R3	
	05	A3	1E	A4	51	611	--		
						612	EXTZV	#TTSV_DS_DTR,#1,UCBSB_TT_DS_TX(R5),R1	: GET CSR ADDRESS
						613			
						614	ASHL	UCBSW_UNIT(R5),R1,R1	: GET CURRENT DTR FOR LINE
						615	BICB	UCBSW_TT_UNITBIT(R5),CRBSB_DZ_DTR(R4)	: SHIFT TO RELATIVE LINE POSITION
							POPL	R4	: RESET CURRENT DTR FOR THAT LINE
							RSB		: SET IT IF NEED BE
									: UPDATE DTR STATUS FOR LINES

IF NDF DZ32SDS_SET:

52	0125	C5	54	DD	02ED	616	PUSHL	R4	: SAVE
	52	F8	52	88	02EF	617	BISB	R2_UCBSB_TT_DS_TX(R5)	: SET NEW OUTPUT SIGNALS
	0125	C5	52	8A	02F4	618	ASHL	#-8,R2,R2	: ACCESS SIGNALS TO RESET
	54	24	A5	DO	02F9	619	BICB	R2_UCBSB_TT_DS_TX(R5)	: RESET THEM
	53	2C	B4	DO	0302	620	MOVL	UCBSL_CRB(R5),R4	: GET CRB ADDRESS
						621	MOVL	ACRBSE_INTD+V ^E CSL_IDB(R4),R3	
						622	TIMEWAIT	#100,DZLC1SM ACK,4(R3),W..TRUE.	: WAIT FOR READY
						623	MOVZWL	UCBSB_TT_DS_TX-T(R5),-(SP)	: CREATE TEMP LOCATION
	7E	0124	C5	3C	0330	624	MOVB	UCBSW_UNIT(R5),(SP)	: SET UNIT NUMBER
	6E	54	A5	90	0335	625	BISW	#DZLC1SM ACK,(SP)	: ENABLE NEW OUTPUT SIGNALS
	6E	8000	8F	A8	0339	626	CVTLW	(SP)+,4(R3)	: SET NEW OUPUT MODEM SIGNALS
	04	A3	8E	F7	033E	627	POPL	R4	
						628	RSB		
						629	.ENDC		

0346 631 .SBTTL DZ-11 MODEM POLLER
 0346 632 ++
 0346 633 DZSTIMER - POLL FOR DZ-11 MODEM TRANSITIONS
 0346 634
 0346 635
 0346 636
 0346 637
 0346 638
 0346 639
 0346 640
 0346 641
 0346 642
 0346 643
 0346 644
 0346 645
 0346 646
 0346 647
 0346 648
 0346 649
 0346 650
 0346 651
 0346 652 DZSTIMER:
 54 0060 8F BB 0346 653 PUSHR #^M<R5,R6>
 54 00000000'EF DE 034A 654 MOVAL DZSL_DIALUP,R4 ; GET DZ TIMER LIST HEAD
 54 64 D0 0351 655 SS: MOVL (R4),R4 ; GET NEXT CRB ADDRESS
 05 12 0354 656 BNEQ 158 ; PROCESS LINES FOR THIS CRB
 0060 8F BA 0356 657 POPR #^M<R5,R6> ; RESTORE REGISTERS
 05 035A 658 RSB ; RETURN FROM TIMER INTERRUPT
 035B 659
 035B 660
 035B 661 TEST LINES ON THIS CONTROLLER FOR A TRANSITION
 035B 662
 035B 663
 035B 664 158: PUSHL R4 ; SAVE TIMER THREAD
 06 53 54 54 54 18 DD 035B 665 SUBL #CRBSL_DZ_MODEM,R4 ; GET ACTUAL CRB ADDRESS
 2C B4 D0 0350 666 MOVL ACRBSL_INTD+VEC\$L_IDB(R4),R3 ; GET CSR ADDRESS
 1C A4 B1 0360 667 CMPW CRBSB_DZ_RING(R4),6(R3) ; ANY TRANSITIONS
 78 13 0364 668 BEQL 608 ; NONE
 0369 669
 036B 670
 036B 671 FIND WHICH SIGNALS CHANGED AND UPDATE THEM
 036B 672
 036B 673
 50 52 06 A3 90 036B 673 MOVBL 6(R3),R2 ; GET NEW RING
 1C A4 52 8D 036F 674 XORB3 R2,CRBSB_DZ_RING(R4),R0 ; FIND TRANSITIONED LINES
 1C A4 52 90 0374 675 MOVBL R2,CRBSB_DZ_RING(R4) ; UPDATE CURRENT RING
 52 07 A3 90 0378 676 MOVBL 7(R3),R2 ; GET NEW CARRIER
 56 1D A4 52 8D 037C 677 XORB3 R2,CRBSB_DZ_CARRIER(R4),R6
 50 56 88 0381 678 BISB R6,R0 ; FLAG LINES WITH TRANSITIONED CARRIER
 1D A4 52 90 0384 679 MOVBL R2,CRBSB_DZ_CARRIER(R4) ; UPDATE CURRENT CARRIER
 0388 680
 0388 681
 0388 682
 0388 683
 51 50 08 00 EA 0388 684 50S: FFS #0,#8,R0,R1 ; FIND NEXT LINE NEEDING SERVICE
 54 13 038D 685 BEQL 608 ; DONE
 00 50 51 E5 038F 686 BBCC R1,R0,558 ; RESET ATTENTION BIT FOR THIS LINE
 0393 687 55S:

56 2C A4	00 0393	688	MOVL	CRBSL_INTD+VEC\$L_IDB(R4),R6
55 18 A641	00 0397	689	MOVL	IDBSL_UCBLST(R6)[R1],R5 : GET UCB FOR THAT LINE
EA	13 039C	690	BEQL	50\$: NONE
E5 44 A5	15 E1	039E 691	BBC	#TTSV_MODEM,UCBSL_DEVDEPEND(R5) 50\$: SKIP IF NOT MODEM LINE
	03A3	692	DSBINT	UCBSB_DIPL(R5) : RAISE TO DEVICE IPL
56 1C A4	01 51	EF 03AA	EXTZV	R1,#1-CRBSB_DZ_RING(R4),R6 : GET RING FOR THAT LINE
01 06	56	FO 03B0	INSV	R6,#TTSV_DS_RING,#1,- : UPDATE IT IN UCB
0124	C5	03B4	UCBSB_TT_DS_RCV(R5)	
56 1D A4	01 51	EF 03B7	EXTZV	R1,#1-CRBSB_DZ_CARRIER(R4),R6 : GET CD FOR THAT LINE
01 05	56	FO 03BD	INSV	R6,#TTSV_DS_CARRIER,#1,- : UPDATE IT IN UCB
0124	C5	03C1	UCBSB_TT_DS_RCV(R5)	
	90 8F	88 03C4	BISB	#<TTSM_DS_DSR!TTSM_DS_CTS>,- : ASSUME CTS AND DSR ALWAYS SET
0124	C5	90 03CA	UCBSB_TT_DS_RCV(R5)	
52 51 03	9A 03CF	703	MOVBL	UCBSB_TT_DS_RCV(R5).R2 : GET CURRENT RECV MODEM STATUS
56 0114	C5	DO 03D2	MOVBL	#MODEMSC_DATASET,R1 : SIGNAL DATASET TRANSITION
	1F BB	03D7	MOVL	UCBSL_TT_CLASS(R5),R6 : GET CLASS VECTOR TABLE
OC B6	16 03D9	706	PUSHR	#^M<R0,RT,R2,R3,R4> : SAVE VOLITAL REGISTERS
1F BA	03DC	707	JSB	QCLASS_DS_TRAN(R6) : SIGNAL TRANSITION
	03DE	708	POPR	#^M<R0,R1,R2,R3,R4> : RESTORE REGISTERS
A5 11	03E1	709	ENBINT	
	03E3	710	BRB	50\$: RESTORE IPL
	03E3	711		60\$:
54 8ED0	03E3	712	POPL	
FF68 31	03E6	713	BRW	R4 : RESTORE TIMER THREAD
	03E9	714		
	03E9	715		

03E9 717 .SBTTL RECEIVER INTERRUPT SERIVCE
 03E9 718 ++
 03E9 719 DZ\$INTINP - DZ RECEIVER READY INTERRUPTS
 03E9 720
 03E9 721 FUNCTIONAL DESCRIPTION:
 03E9 722 THIS ROUTINE IS ENTERED WHEN A CHARACTER IS AVAILABLE IN THE UNIT'S
 03E9 723 SILO. THE CHARACTER IS EXTRACTED AND IS PASSED TO THE ASSOCIATED
 03E9 724 CLASS DRIVER. IF THE CLASS DRIVER RETURNS CHARACTERS(S) THEN NEW
 03E9 725 OUTPUT IT INITIATED (NORMALLY ECHO).
 03E9 726
 03E9 727
 03E9 728 INPUTS:
 03E9 729
 03E9 730 00(SP) = ADDRESS OF IDB
 03E9 731
 03E9 732 IMPLICIT INPUTS:
 03E9 733
 03E9 734 R0,R1,R2,R3,R4,R5 ARE SAVED ON STACK.
 03E9 735
 03E9 736 OUTPUTS:
 03E9 737
 03E9 738 THE INTERRUPT IS DISMISSED WHEN THE SILO IS EMPTY.
 03E9 739
 03E9 740 --
 03E9 741 DZ11\$INTINP:: : DZ-11 INPUT INTERRUPTS
 03E9 742
 03E9 743 GET THE CSR ADDRESS
 03E9 744
 54 9E DD 03E9 745 MOVL a(SP)+,R4 : GET THE IDB ADDRESS
 54 DD 03EC 746 PUSHL R4 : SAVE IDB ADDRESS
 50 64 DD 03EE 747 MOVL (R4),R0 : GET THE CSR ADDRESS
 03F1 748
 03F1 749 GET THE CHARACTER FROM THE INTERFACE
 03F1 750
 53 02 A0 B0 03F1 751 25\$: MOVW 2(R0),R3 : GET THE CHARACTER,ERRORS AND LINE NUMBER
 73 18 03F5 752 BGEQ 100\$: SILO EMPTY
 53 7000 8F B3 03F7 753 BITW #<CDZRCVSM_PARERR>!-
 03FC 754 <CDZRCVSM_OVERRUN>!-
 03FC 755 <CDZRCVSM_FRAMER>,R3 : ERRORS?
 52 53 F8 8F 48 12 03FC 756 BNEQ 50\$: YES PROCESS THEM
 52 FFFFFFFF 8F 78 03FE 757 27\$: ASHL #-8,R3,R2 : GET THE LINE NUMBER
 53 53 9A 0403 758 BICL #^C<7>,R2
 55 18 A442 DD 0400 760 MOVBL R3,R3 : CLEAR THE HIGH BYTES OF CHARACTER
 DD 13 0412 761 BEQL IDBSL_UCBLST(R4)[R2],R5 : GET THE UCB FOR THAT LINE
 0110 D5 16 0414 762 JSB UCBSL_TT_PUTNXT(R5) : IF EQL THEN NOT THERE
 0108 C5 95 0418 763 TSTB UCBSB_TT_OUTYPE(R5) : BUFFER THE CHARACTER
 17 15 041C 764 BLEQ 40\$: DID HE RETURN ANYTHING TO OUTPUT
 0108 C5 90 041E 765 MOVB R3_UCBSW_TT_HOLD(R5) : NONE OR STRING OUTPUT
 0400 8F AB 0422 766 BISW #TTYSM_TANK_HOLD,- : SAVE THE CHARACTER IN TANK
 0108 C5 0427 767 UCBSW_TT_HOLD(R5) : SIGNAL CHARACTER IN TANK
 04 A0 0106 C5 AB 042A 768 BISW UCBSW_TT_UNITBIT(R5),4(R0) : ENABLE LINE
 54 6E DD 0430 769 30\$: MOVL (SP),R4 : GET IDB ADDRESS
 BC 11 0433 770 BRB 25\$: CONTINUE
 0800 8F AB 0435 771 40\$: BEQL 30\$: NO CHARACTER
 0437 772 BISW #TTYSM_TANK_BURST,- : SIGNAL BURST

04 A0	0108 C5	043B	774					
	0106 C5	A8	043E	775	BISW	UCBSW_TT_HOLD(R5)		
	EA	11	0444	776	BRB	UCBSW_TT_UNITBIT(R5),4(R0)		: ENABLE LINE
			0446	777		30\$		
			0446	778				: SILO EMPTY OR CHARACTER IN ERROR
			0446	779				
			0446	780	50\$:			
			0446	781				
			0446	782				: PROCESS PARITY, FRAME OR OVERRUN ERROR
			0446	783				
52	53 F8 8F	78	0446	784	ASHL	#-8,R3,R2		: GET LINE NUMBER
52	FFFFFFFFFF8 8F	CA	044B	785	BICL	#^C<7>,R2		
55	18 A442	D0	0452	786	MOVL	IDBSL_UCBLST(R4)[R2],R5;	GET UCB ADDRESS	
52	0A	13	0457	787	BEQL	70\$: IF EQL THEN NO UCB
52	0114 C5	D0	0459	788	MOVL	UCBSL_TT_CLASS(R5),R2		: GET CLASS DISPATCH
14	B2	16	045E	790	JSB	ACCLASS_READERROR(R2)		: SIGNAL ERROR
60	0080 8F	9B	0461	791	BNEQ	27\$		
		B3	0463	792	BITW	#^X080,(R0)		: BRANCH WITH CHARACTER TO MAIN PATH
		C6	0468	793	BNEQ	30\$: VALID CHARACTER IN SILO NOW?
SE	04	C0	046A	794	100\$:	ADDL		: IF NEQ THEN YES
50	8E	7D	046D	795	MOVQ	#4,SP		: REMOVE IDB ADDRESS
52	8E	7D	0470	796	MOVQ	(SP)+,R0		: RESTORE REGISTERS
54	8E	7D	0473	797	MOVQ	(SP)+,R2		
		02	0476	798	MOVQ	(SP)+,R4		
			0477	799	REI			

0477 801 : IF NDF DZV
 0477 802 :
 0477 803 : DZ-32 INPUT INTERRUPT SERVICE
 0477 804 :
 0477 805 :
 0477 806 DZ32\$INTINP:: : DZ-32 INPUT INTERRUPTS
 0477 807 :
 0477 808 : GET THE CSR ADDRESS
 0477 809 :
 54 9E DD 0477 810 MOVL a(SP)+,R4 : GET THE IDB ADDRESS
 54 DD 047A 811 PUSHL R4 : SAVE IDB ADDRESS
 50 64 DO 047C 812 MOVL (R4),R0 : GET THE CSR ADDRESS
 AA 047F 813 BICW #<>DZCSRSM_RCVINT>!- : DISABLE RECEIVER INTERRUPTS
 60 0042 8F 0480 814 <DZCSRSM_D5_ENAB>,- : DISABLE DZ-32 DATA SET INTERRUPTS
 0480 815 (R0) : DZ-32 during the interrupt service routine
 0484 816 :
 0484 817 : GET THE CHARACTER FROM THE INTERFACE
 53 02 A0 80 0484 818 25\$: MOVW 2(R0),R3 : GET THE CHARACTER,ERRORS AND LINE NUMBER
 75 18 0488 819 BGEQ 100\$: SILO EMPTY
 53 7000 8F 83 048A 820 BITW #<>DZRCVSM_PARERR>!- : ERRORS?
 048F 821 <DZRCVSM_OVERRUN>!- : ERRORS?
 048F 822 <DZRCVSM_FRAMER>,R3 : ERRORS?
 52 53 F8 8F 4A 12 048F 824 27\$: BNEQ 50\$: YES, PROCESS THEM
 52 FFFFFFF8 8F 78 0491 825 ASHL #>-8,R3,R2 : GET THE LINE NUMBER
 53 53 CA 0496 826 BICL #>C<7>,R2 :
 55 18 A442 DO 04A0 828 MOVZBL R3,R3 : CLEAR THE HIGH BYTES OF CHARACTER
 DD 13 04A5 829 MOVL IDBSL_UCBLST(R4)[R2],R5 : GET THE UCB FOR THAT LINE
 0110 D5 16 04A7 830 BEQL 25\$: IF EQL THEN NOT THERE
 0108 C5 95 04AB 831 JSB UCBSL_TT_PUTNXT(R5) : BUFFER THE CHARACTER
 1F 15 04AF 832 TSTB UCBSB_TT_OUTYPE(R5) : DID HE RETURN ANYTHING TO OUTPUT
 0108 C5 90 04B1 833 BLEQ 40\$: NONE OR STRING OUTPUT
 0400 8F A8 04B6 834 MOVB R3,UCBSW_TT_HOLD(R5) : SAVE THE CHARACTER IN TANK
 0108 C5 04BA 835 BISW #>TYSM_TANK_HOLD,- : SIGNAL CHARACTER IN TANK
 04BD 836 28\$: UCBSW_TT_HOLD(R5)
 54 6E DD 04BD 837 MOVL (SP),R4 : RESTORE IDB ADDRESS
 0106 C5 88 04C0 838 BISB UCBSW_TT_UNITBIT(R5),- : ENABLE LINE
 OE A4 04C4 839 IDBSB_TT_ENABLE(R4)
 07 A0 DE A4 90 04C6 840 MOVB IDBSB_TT_ENABLE(R4),7(R0)
 54 6E DD 04CB 841 30\$: MOVL (SP),R4 : GET IDB ADDRESS
 B4 11 04CE 842 BRB 25\$: CONTINUE
 04D0 843 40\$: BEQL 30\$: NO CHARACTER
 0800 8F AB 04D2 845 BISW #>TYSM_TANK_BURST,- : SIGNAL BURST
 0108 C5 04D6 846 UCBSW_TT_HOLD(R5)
 E2 11 04D9 847 BRB 28\$:
 04DB 848 : SILO EMPTY OR CHARACTER IN ERROR
 04DB 849 :
 04DB 850 :
 04DB 851 :
 04DB 852 50\$: :
 04DB 853 :
 04DB 854 : PROCESS PARITY, FRAME OR OVERRUN ERROR
 04DB 855 :
 52 53 F8 8F 78 04DB 856 ASHL #>-8,R3,R2 : GET LINE NUMBER
 52 FFFFFFF8 8F CA 04E0 857 BICL #>C<7>,R2 :

55	18	A442	D0	04E7	858		MOVL	IDBSL_UCBLST(R4)[R2],R5	: GET UCB ADDRESS
52	0114	C5	D0	04EF	859		BEQL	?0\$: IF EQL THEN NO UCB
				04F3	860		MOVL	UCBSL_TT_CLASS(R5),R2	: GET CLASS DISPATCH
	14	B2	16	04F3	862	60\$:	JSB	ACCLASS_READERROR(R2)	: SIGNAL ERROR
	99	12	04F6	863	864	70\$:	BNEQ	27\$: BRANCH WITH CHARACTER TO MAIN PATH
60	0080	8F	B3	04F8	864	70\$:	BITW	#^X080,(R0)	: VALID CHARACTER IN SILO NOW?
	CC	12	04FD	865	866	100\$:	BNEQ	30\$: IF NEQ THEN YES
	53	60	B0	04FF	867		MOVW	(R0),R3	: TEST FOR MODEM TRANSITION
12	53	08	E0	0502	868		BBS	#DZCSRSM_DS_CHG,R3,200\$: BRANCH IF MODEM TRANSITION
			A8	0506	869		BISW	#<<DZCSRSM_RCVINT>!- <DZCSRSM_DS_ENAB>>,-	: ENABLE RECEIVER INTERRUPTS
60	0042	8F		0507	870		ADDL	(R0)	: ENABLE DZ-32 DATA SET INTERRUPTS
5E	04	C0		0508	871		#4,SP		: DZ-32 BEFORE EXITING
50	8E	7D		050E	872		MOVQ	(SP)+,R0	: REMOVE IDB ADDRESS
52	8E	7D		0511	873		MOVQ	(SP)+,R2	: RESTORE REGISTERS
54	8E	7D		0514	874		MOVQ	(SP)+,R4	
				02	875		REI		
		50	DD	0518	876	200\$:	PUSHL	R0	: SAVE R0
52	53	06	A0	90	051A	877	MOVB	6(R0),R3	: GET NEW RECEIVE MODEM SIGNALS
	FFFFFFFFFF8	8F	CB	051E	878		BICL3	#^C<7>,R3,R2	: ISOLATE UNIT NUMBER
55	18	A442	D0	0526	879		MOVL	IDBSL_UCBLST(R4)[R2],R5	: GET ASSOCIATED UCB
	13	13	052B	880			BEQL	110\$: NONE
0124	C5	53	90	052D	881		MOVB	R3,UCBSB_TT_DS_RCV(R5)	: UPDATE INPUT MODEM SIGNALS
	52	53	9A	0532	882		MOVZBL	R3,R2	: LOAD ARGUMENT
	51	03	9A	0535	883		MOVZBL	#MODEMSC_DATASET,R1	: MODEM TRANSITION
53	0114	C5	D0	0538	884		MOVL	UCBSL_TT_CLASS(R5),R3	: ACCESS CLASS VECTORS
	OC	B3	16	053D	885		JSB	ACCLASS_DS_TRAN(R3)	: SIGNAL TRANSITION
	50	8ED0		0540	886		POPL	R0	: RESTORE R0
FF85		31		0543	887	110\$:	BRW	30\$: DISMISS INTERRUPT
				0546	888				
				0546	889	.ENDC			

0546 891 .SBTTL START I/O ROUTINE
 0546 892 ++
 0546 893 DZ\$STARTIO - START I/O OPERATION ON DZ
 0546 894
 0546 895
 0546 896
 0546 897
 0546 898
 0546 899
 0546 900
 0546 901
 0546 902
 0546 903
 0546 904
 0546 905
 0546 906
 0546 907
 0546 908
 0546 909
 0546 910 --
 0546 911 DZ11\$STARTIO:: :
 0800 16 18 0546 912 BGEQ 20S : START I/O ON UNIT
 0108 C5 A8 0548 913 BISW #TTYSM_TANK_BURST,- : SINGLE CHARACTER
 51 24 A5 D0 054C 914 UCBSW_TT_HOLD(R5) : SIGNAL BURST ACTIVE
 51 2C B1 D0 0553 915 10\$: MOVL UCBSL_CRB(R5),R1 : GET CRB OF UNIT
 04 A1 0106 C5 A8 0557 916 MOVL @CRBSL_INTD+VECSL_IDB(R1),R1 : GET CSR
 05 055D 917 BISW UCBSW_TT_UNITBIT(R5),4(R1) ; ENABLE LINE
 RSB 918 : RETURN TO CALLER
 0108 C5 53 90 055E 919 20\$: MOVB R3,UCBSW_TT_HOLD(R5) : SAVE OUTPUT CHARACTER
 0400 8F A8 0563 920 BISW #TTYSM_TANK_HOLD,- : SIGNAL CHARACTER IN TANK
 0108 C5 E3 11 0567 921 UCBSW_TT_HOLD(R5)
 056A 922 BRB 10\$
 056C 923
 056C 924
 056C 925 .IF NDF DZV
 056C 926 DZ32\$STARTIO:: :
 0800 1E 18 056C 927 BGEQ 20S : START I/O ON UNIT
 0108 C5 A8 056E 928 BISW #TTYSM_TANK_BURST,- : SINGLE CHARACTER SPECIFIED
 51 24 A5 D0 0575 929 UCBSW_TT_HOLD(R5) : SIGNAL BURST ACTIVE
 54 2C A1 D0 0579 930 10\$: MOVL UCBSL_CRB(R5),R1 : GET CRB OF UNIT
 51 64 D0 057D 931 MOVL CRBSL_INTD+VECSL_IDB(R1),R4 : GET IDB ADDRESS
 0E A4 0106 C5 88 0580 932 MOVL (R4),R1 : GET CSR ADDRESS
 07 A1 0E A4 90 0586 933 BISB UCBSW_TT_UNITBIT(R5),IDBSB_TT_ENABLE(R4)
 05 0588 934 MOVB IDBSB_TT_ENABLE(R4),7(R1)
 RSB 935 : RETURN TO CALLER
 0108 C5 53 90 058C 936 20\$: MOVB R3,UCBSW_TT_HOLD(R5) : SAVE OUTPUT CHARACTER
 0400 8F A8 0591 937 BISW #TTYSM_TANK_HOLD,- : SIGNAL CHARACTER IN TANK
 0108 C5 DB 11 0595 938 UCBSW_TT_HOLD(R5)
 0598 940 BRB 10\$
 059A 941 .ENDC

```

059A 943 .SBTTL PORT ROUTINES STOP,RESUME,XON,XOFF
059A 944 ++
059A 945 DZ$XOFF - SEND XOFF
059A 946 DZ$XON - SEND XON
059A 947 DZ$STOP - STOP OUTPUT
059A 948 DZ$ABORT - ABORT CURRENT OUTPUT
059A 949 DZ$RESUME - RESUME STOPPED OUTPUT
059A 950
059A 951 FUNCTIONAL DESCRIPTION:
059A 952
059A 953 THESE ROUTINES ARE USED BY THE THE TERMINAL CLASS DRIVER TO
059A 954 CONTROL OUTPUT ON THE PORT
059A 955
059A 956 INPUTS:
059A 957
059A 958 RS = UCB ADDRESS
059A 959
059A 960 OUTPUTS:
059A 961
059A 962 RS = UCB ADDRESS
059A 963 --
059A 964 .ENABLE LSB
059A 965
059A 966 SCHEDULE XOFF TO BE SENT
059A 967
059A 968 INPUTS:
059A 969 R3 - CHARACTER TO BE SENT AS FLOW CONTROL
059A 970
059A 971 DZ11$XOFF:
059A 972
059A 973 : SCHEDULE XON TO BE SENT
059A 974
059A 975 DZ11$XON:
0108 C5 0100 8F A8 059A 976 BISW #TTYSM_TANK_PREMPT,UCBSW_TT_HOLD(R5) : SCHEDULE XON
010A C5 53 90 05A1 977 MOVB R3,UCBSB_TT_PREMPT(R5) : SAVE THE CHARACTER
05A6 978
18 64 A5 01 E0 05A6 979 BBS #UCBSV_INT,UCBSW_STS(R5),10$ : IF OUTPUT ACTIVE,
05A6 980
05AB 981 PUSHL R1 : FINISHED
51 24 A5 DD 05AD 982 MOVL UCBSL(CRB(R5),R1 : SAVE A REGISTER
51 2C B1 DD 05B1 983 MOVL ACRBSL INTD+VÉ(SL_IDB(R1),R1 : ACCESS CRB ADDRESS
04 A1 0106 C5 A8 05B5 984 BISW UCBSW_TT_UNITBIT(R5),4(R1) : GET CSR ADDRESS
51 8ED0 05B8 985 POPL R1 : ENABLE LINE
00 64 A5 01 E2 05BE 986 BBSS #UCBSV_INT,UCBSW_STS(R5),10$ : SHOW OUTPUT ACTIVE
05C3 987 10$: RSB
05C3 988 .DISABLE LSB
05C4 989
05C4 990 : STOP PORT OUTPUT
05C4 991
05C4 992 : DZ$STOP:
05C4 993
0200 8F A8 05C4 994 BISW #TTYSM_TANK_STOP - : SCHEDULE STOP
0108 C5 05C8 995 UCBSW_TT_HOED(R5)
05 05CB 996 RSB
05CC 997
05CC 998 : ABORT ANY CURRENT PORT OUTPUT ACTIVITY
05CC 999

```


06DC 1085 .SBTTL OUTPUT INTERRUPT SERVICE
 06DC 1086 ++
 06DC 1087 DZSINTOUT - DZ-11 OUTPUT INTERRUPT SERVICE
 06DC 1088
 06DC 1089 : FUNCTIONAL DESCRIPTION:
 06DC 1090
 06DC 1091 : THIS ROUTINE IS ENTERED WHEN THE DZ-11 FINDS A LINE ENABLED
 06DC 1092 : AND AN EMPTY UART. THE CORRESPONDING UCB IS FOUND AND
 06DC 1093 : ANY OUTSTANDING PORT OUTPUT IS DONE. WHEN ALL OUTSTANDING PORT
 06DC 1094 : OUTPUT IS COMPLETED, THE CLASS DRIVER IS CALLED TO RETURN THE NEXT
 06DC 1095 : CHARACTER OR STRING TO BE OUTPUT. IF NO MORE OUTPUT IS FOUND, THEN
 06DC 1096 : THE LINE IS DISBALED.
 06DC 1097
 06DC 1098 : INPUTS:
 06DC 1099
 06DC 1100 : SP(00) = ADDRESS OF THE IDB
 06DC 1101
 06DC 1102 : IMPLICIT INPUTS:
 06DC 1103
 06DC 1104 : R0,R1,R2,R3,R4,R5 SAVED ON THE STACK.
 06DC 1105
 06DC 1106 : OUTPUTS:
 06DC 1107
 06DC 1108 : THE INTERRUPT IS DISMISSED.
 06DC 1109
 06DC 1110 --
 5E 04 C0 06DC 1111 DZ11_OUT EXIT:
 50 8E 7D 06DC 1112 ADDL #4,SP
 52 8E 7D 06DF 1113 MOVQ (SP)+,R0
 54 8E 02 06E2 1114 MOVQ (SP)+,R2
 06E5 1115 MOVQ (SP)+,R4
 06E8 1116 REI
 06E9 1117
 06E9 1118 DZ11SINTOUT:: : EXIT OUTPUT INTERRUPT
 06E9 1119 : REMOVE IDB ADDRESS
 06E9 1120 DZ11_OUT LOOP: : RESTORE REGISTERS
 54 00 BE DD 06E9 1121 MOVL @SP,R4 : DISMISS INTERRUPT
 50 64 DD 06ED 1122 MOVL (R4),R0 : DZ-11 OUTPUT INTERRUPT SERVICE
 06F0 1123 : GET THE IDB ADDRESS
 06F0 1124 : GET THE CSR ADDRESS
 06F0 1125 : GET THE LINE INFO FROM THE CSR
 06F0 1126
 52 60 B0 06F0 1127 MOVW (R0),R2 : GET THE CSR VALUE
 E7 18 06F3 1128 BGEQ DZ11_OUT_EXIT : NO MORE LINES
 52 F8 8F 06F5 1129 ASHL #-8,R2,R2 : GET THE LINE NUMBER
 FFFFFFF8 8F CA 06FA 1130 BICL #^C<7>,R2
 55 18 A442 DO 0701 1131 MOVL IDBSL_UCBLST(R4)[R2],R5 : GET THE UCB ADDRESS
 E1 13 0706 1132 BEQL DZ11_OUT_LOOP : IF EQL THEN DISMISS
 0708 1133 :
 0708 1134 : CHECK FOR BURST ACTIVE ON LINE
 0708 1135 :
 0109 08 91 0708 1136 CMPB #TTYSM_TANK_BURST@-8,- : ONLY BURST ACTIVE?
 C5 40 13 070A 1137 UCBSW TT_HO[D+1](R5)
 070D 1138 BEQL DZ11_BURST : YES, CONTINUE BURST
 070F 1139 :
 070F 1140 : LOOK FOR NEXT OUTPUT STATE IN TANK
 070F 1141 :

53 0109 C5 06 00 EA 070F 1142
 070F 1143
 0716 1144 FFS CASE #0,#6,UCBSW_TT_HOLD+1(R5),R3
 0716 1145 R3_TYPE=B,<=
 0716 1146 DZ11_PREMPT,-
 0716 1147 DZ11_STOP,-
 0716 1148 DZ11_CHAR,-
 0716 1149 DZ11_BURST,-
 0722 1150 >
 0722 1151 : NO PENDING DATA - LOOK FOR NEXT CHARACTER
 0722 1152 :
 64 A5 03 BA 0722 1153 BICB #UCBSM_TIM!UCBSM_INT,UCBSW_STS(R5); CLEAR TIMEOUT AND EXPECTED
 0726 1154 :
 0726 1155 : CALL CLASS DRIVER FOR MORE OUTPUT
 0726 1156 :
 01 FF BF 010C D5 16 0726 1157 JSB @UCBSL_TT_GETNXT(R5) : GET THE NEXT CHARACTER
 010B C5 8F 072A 1158 CASEB UCBSB_TT_OUTTYPE(R5),#-1,#1: OPTIMIZE FOR THE SINGLE
 0731 1159 : CHARACTER CASE BY SETTING THE
 0731 1160 : LIMIT TO 1
 0017' 0731 1161 1\$: .WORD DZ11_START_BURST-1\$: BURST SPECIFIED
 000A' 0733 1162 .WORD 50S-TS : NONE
 0735 1163 :
 0735 1164 : OUTPUT A CHARACTER TO THE DZ-11
 0735 1165 :
 06 A0 53 AE 9B 0735 1166 20\$: MOVZBW R3,6(R0) ; OUTPUT CHARACTER
 0739 1167 BRB DZ11_OUT_LOOP
 073B 1168 : DISABLE OUTPUT ON THIS LINE
 073B 1169 :
 073B 1170 :
 073B 1171 50\$:
 A9 64 A5 01 E0 073B 1172 BBS #UCBSV_INT,- : IF INT EXP, THEN DON'T RESET,
 073D 1173 UCBSW_STS(R5),DZ11_OUT_LOOP : COULD HAVE BEEN SET DURING CALLBACK
 0740 1174 :
 0740 1175 :
 04 A0 0106 C5 A1 AA 0740 1176 BICW UCBSW_TT_UNITBIT(R5),4(R0) ; RESET THE OUTPUT ENABLE
 0746 1177 BRB DZ11_OUT_LOOP
 0748 1178 :
 0748 1179 :
 0800 BF 0108 C5 A8 0748 1180 DZ11_START_BURST:
 0748 1181 BISW #TTYSM_TANK_BURST,- ; SIGNAL BURST ACTIVE
 074C 1182 UCBSW_TT_HOLD(R5)
 074F 1183 :
 074F 1184 : CONTINUE BURST OUTPUT
 074F 1185 :
 074F 1186 DZ11_BURST:
 011C D5 90 074F 1187 MOVB UCBSL_TT_OUTADR(R5),- ; OUTPUT NEXT BYTE
 06 A0 0753 1188 6(R0)
 011C C5 D6 0755 1189 INCL UCBSL_TT_OUTADR(R5)
 0120 C5 B7 0759 1190 DECW UCBSW_TT_OUTLEN(R5) ; UPDATE POINTER
 8A 12 075D 1191 BNEQ DZ11_OUT_LOOP ; UPDATE COUNT
 0800 BF AA 075F 1192 BICW #TTYSM_TANK_BURST,- ; NOT LAST CHARACTER
 0108 C5 0763 1193 UCBSW_TT_HOLD(R5) ; RESET BURST ACTIVE
 FF80 31 0766 1194 BRW DZ11_OUT_LOOP
 0769 1195 :
 0769 1196 : OUTPUT SINGLE CHARACTER
 0769 1197 :
 0769 1198 DZ11_CHAR:

06 A0	0108 C5	90	0769	1199		MOV B	UCBSW_TT_HOLD(R5),6(R0) ; OUTPUT CHAR IN TANK
	0400 8F	AA	076F	1200		BIC W	#TTYSM_TANK HOLD - ; SHOW TANK EMPTY
	0108 C5		0773	1201			UCBSW_TT_HOLD(R5)
	FF70	31	0776	1202		BRW	DZ11_OUT_LOOP
			0779	1203			
			0779	1204	:		STOP THE OUTPUT
			0779	1205			
			0779	1206	DZ11_STOP:		
04 A0	64 A5	03	8A	0779	1207	BIC B	#UCBSM_INT!UCBSM_TIM,-
	0106 C5	AA	077B	1208			UCBSW_STS(R5)
	FF63	31	077D	1209		BIC W	UCBSW_TT_UNITBIT(R5),4(R0) ; RESET OUTPUT ACTIVE
			0783	1210		BRW	DZ11_OUT_LOOP ; RESET THE OUTPUT ENABLE
			0786	1211			
			0786	1212			.ENABLE LSB
			0786	1213			
			0786	1214	:		SEND XON OR XOFF CHARACTER
			0786	1215			
			0786	1216			
			0786	1217	DZ11_PREMPT:		
06 A0	0100 8F	AA	0786	1218		BIC W	#TTYSM_TANK PREMPT,- ; RESET XOFF STATE
	0108 C5		078A	1219			UCBSW_TT_HOLD(R5)
	010A C5	90	078D	1220		MOV B	UCBSB_TT_PREMPT(R5),6(R0); OUTPUT CHARACTER
	FF53	31	0793	1221		BRW	DZ11_OUT_LOOP
			0796	1222			.DISABLE [SB
			0796	1223			

0796 1225 .IF NDF DZV
 0796 1226 :
 0796 1227 : DZ-32 OUTPUT INTERRUPT SERVICE CODE
 0796 1228 :
 0796 1229 :
 SE 04 CO 0796 1230 DZ32_OUT_EXIT:
 50 8E 7D 0799 1231 ADDL #4, SP
 52 8E 7D 079C 1232 MOVQ (SP)+, R0
 54 8E 7D 079F 1233 MOVQ (SP)+, R2
 02 07A2 1234 MOVQ (SP)+, R4
 07A3 1235 REI
 07A3 1236 :
 54 00 BE DO 07A3 1237 DZ32\$INTOUT::
 50 50 64 DO 07A3 1238 DZ32_OUT_LOOP:
 07A7 1239 MOVL a(SP), R4
 07AA 1240 MOVL (R4), R0
 07AA 1241 : GET THE LINE INFO FROM THE CSR
 07AA 1242 :
 07AA 1243 :
 07AA 1244 :
 52 60 B0 07AA 1245 MOVW (R0), R2
 52 52 F8 8F E7 18 07AD 1246 BGEQ DZ32_OUT_EXIT
 52 FFFFFFF8 8F CA 07AF 1247 ASHL #-8, R2, R2
 55 18 A442 E1 13 07B4 1248 BICL #^C<7>, R2
 07B8 1249 MOVL IDBSL_UCBLST(R4)[R2], R5
 07C0 1250 BEQL DZ32_OUT_LOOP ; GET THE UCB ADDRESS
 07C2 1251 : IF EQL THEN DISMISS
 07C2 1252 : CHECK FOR BURST ACTIVE ON LINE
 07C2 1253 :
 0109 08 91 07C2 1254 CMPB #TTYSM_TANK_BURST-8,- ; ONLY BURST ACTIVE?
 07C4 1255 UCB\$W TT HOLD+1(R5)
 4B 13 07C7 1256 BEQL DZ32_BURST ; YES, CONTINUE BURST
 07C9 1257 :
 07C9 1258 : LOOK FOR NEXT OUTPUT STATE IN TANK
 07C9 1259 :
 07C9 1260 :
 53 0109 C5 06 00 EA 07C9 1261 FFS CASE #0, #6, UCB\$W_TT_HOLD+1(R5), R3
 07D0 1262 R3_TYPE=B, <= DISPATCH
 07D0 1263 DZ32_PREMPT,- SEND PREMPT CHARACTERS
 07D0 1264 DZ32_STOP,- STOP OUTPUT
 07D0 1265 DZ32_CHAR,- CHAR IN TANK
 07D0 1266 DZ32_BURST,- BURST IN PROGRESS
 07D0 1267 >
 07DC 1268 : NO PENDING DATA - LOOK FOR NEXT CHARACTER
 07DC 1269 :
 64 A5 03 8A 07DC 1270 :
 07E0 1271 BICB #UCBSM_TIM!UCBSM_INT, UCB\$W_STS(R5); CLEAR TIMEOUT AND EXPECTED
 07E0 1272 :
 07E0 1273 : CALL CLASS DRIVER FOR MORE OUTPUT
 07E0 1274 :
 01 FF 8F 010C D5 16 07E0 1275 JSB UCBSL_TT_GETNXT(R5) ; GET THE NEXT CHARACTER
 010B C5 8F 07E4 1276 CASEB UCBSB_TT_OUTTYPE(R5), #1-#1; OPTIMIZE FOR THE SINGLE
 07EB 1277 : CHARACTER CASE BY SETTING THE
 07EB 1278 : LIMIT TO 1
 0022: 07EB 1279 1\$: .WORD DZ32_START_BURST-1\$; BURST SPECIFIED
 000B: 07ED 1280 .WORD 508-T\$; NONE
 07EF 1281 :

06 A0 53 FFAD 90 31 07EF 1282 : OUTPUT A CHARACTER TO THE DZ-32
 07EF 1283
 07EF 1284 20\$: MOVB R3,6(R0)
 07F3 1285 BRW DZ32_OUT_LOOP ; OUTPUT CHARACTER
 07F6 1286
 07F6 1287
 07F6 1288 : DISABLE OUTPUT ON THIS LINE
 07F6 1289
 07F6 1290 50\$: BBS #UCBSV_INT,-
 A8 64 01 E0 07F6 1291 UCBSW_STS(R5),DZ32_OUT_LOOP : IF INT EXP, THEN DON'T RESET,
 07F8 1292 ; COULD HAVE BEEN SET DURING CALLBACK
 07FB 1293
 07F8 1294
 54 00 BE 0106 C5 0E A4 07FB 1295 MOVL a(SP),R4
 D0 8A 07FF 1296 BICB UCBSW_TT_UNITBIT(R5),- : GET IDB ADDRESS
 0803 1297 IDBSB_TT_ENABLE(R4)
 07 A0 0E A4 07 A0 0805 1298 MOVB IDBSB_TT_ENABLE(R4),7(R0)
 FF96 31 080A 1299 BRW DZ32_OUT_LOOP
 080D 1300
 080D 1301
 0800 8F A8 080D 1302 DZ32_START_BURST:
 0108 C5 0811 1303 BISW #TTYSM_TANK_BURST,- : SIGNAL BURST ACTIVE
 0814 1304 UCBSW_TT_HOLD(R5)
 011C D5 90 0814 1305 DZ32_BURST:
 06 A0 0818 1306 MOVB #UCBSL_TT_OUTADR(R5),- : OUTPUT NEXT BYTE
 011C C5 081A 1308 INCL 6(R0)
 0120 C5 081E 1309 DECW UCBSL_TT_OUTLEN(R5) : UPDATE POINTER
 07 12 0822 1310 BNEQ 60\$ UCBSW_TT_OUTLEN(R5) : UPDATE COUNT
 0800 8F AA 0824 1311 BICW #TTYSM_TANK_BURST,- : NOT LAST CHARACTER
 0108 C5 0828 1312 UCBSW_TT_HOLD(R5) : RESET BURST ACTIVE
 FF75 31 082B 1313 60\$: BRW DZ32_OUT_LOOP
 082E 1314 :
 082E 1315 : OUTPUT SINGLE CHARACTER
 082E 1316 :
 06 A0 0108 C5 90 082E 1317 DZ32_CHAR:
 0400 8F AA 0834 1318 MOVB UCBSW_TT_HOLD(R5),6(R0) : OUTPUT CHAR IN TANK
 0108 C5 0838 1319 BICW #TTYSM_TANK_HOLD,- : SHOW TANK EMPTY
 FF65 31 083B 1320 UCBSW_TT_HOLD(R5)
 083E 1321 BRW DZ32_OUT_LOOP
 083E 1322 : STOP OUTPUT
 083E 1323 :
 083E 1324 :
 083E 1325 DZ32_STOP:
 54 03 8A 083E 1326 BICB #UCBSM_INT!UCBSM_TIM,-
 64 A5 0840 1327 UCBSW_STS(R5)
 00 BE 0106 C5 0842 1328 MOVL a(SP),R4 : RESET OUTPUT ACTIVE
 D0 8A 0846 1329 BICB UCBSW_TT_UNITBIT(R5),- : GET IDB ADDRESS
 0E A4 084A 1330 IDBSB_TT_ENABLE(R4)
 07 A0 0E A4 084C 1331 MOVB IDBSB_TT_ENABLE(R4),7(R0)
 FF4F 31 0851 1332 BRW DZ32_OUT_LOOP
 0854 1333 .ENABLE LSB
 0854 1334
 0854 1335 : SEND XON OR XOFF
 0854 1336
 0854 1337
 0854 1338 DZ32_PREMPT:

06 A0 0100 8F AA 0854 1339 BICW #TTYSM TANK PREMPT,- ; RESET XOFF STATE
0108 C5 0858 1340 UCB\$W_TT_HOLD(R5)
010A C5 90 085B 1341 MOVB UCB\$B_TT_PREMPT(R5),6(R0); OUTPUT CHARACTER
FF3F 31 0861 1342 BRW DZ32_OUT_LOOP
0864 1343
0864 1344
0864 1345 .DISABLE LSB
0864 1346 .ENDC

0864 1348 .SBTTL SET SPEED, PARITY PARAMETERS

0864 1349

0864 1350 ++

0864 1351 DZ\$SET_LINE - RESET SPEED, PARITY

0864 1352

0864 1353 FUNCTIONAL DESCRIPTION:

0864 1354

0864 1355 INPUTS:

0864 1356 R5 - UCB ADDRESS

0864 1357

0864 1358

0864 1359

0864 1360

0864 1361 OUTPUTS:

0864 1362 -- R4 USED

0864 1363

54 24 A5 D0 0864 1364 DZ\$SET_LINE:

0864 1365 MOVL UCB\$L_CRB(R5),R4 : ADDRESS CRB

0868 1366

0868 1367 SET UP LINE SPEED AND PARITY

0868 1368

54 2C B4 D0 0868 1369

00F4 C5 01 7E D4 086C 1370

6E 00F8 C5 90 086E 1371

6E F007 8F AA 0875 1372

05 012A C5 07 E0 087A 1373

6E 1000 8F A8 087F 1374

0881 1375

0885 1376

088A 1377 38:

6E 54 A5 A8 088A 1378

64 01 B3 088E 1379

05 12 0891 1380

0893 1381 58:

02 A4 8E F7 0893 1382

05 0897 1383

0898 1384

0898 1385

0898 1386 108:

00F4 C5 91 0898 1387

00F5 C5 089C 1388

F2 13 089F 1389

00F5 C5 95 08A1 1390

EC 13 08A5 1391

08A7 1392

08A7 1393 : SET SPLIT SPEED

08A7 1394

6E 2000 8F A8 08A7 1395

E5 11 08AC 1396

08AE 1397

MOVL @CRBSL_INTD+VECSL_IDB(R4),R4 : GET THE CSR ADDRESS VIA CRB

CLRL -(SP) : RESET A TEMPORARY LOCATION

SUBB3 #1,UCBSW_TT_SPEED(R5),1(SP) : ADJUST DATA BASE SPEED

MOVBL UCBSB_TT_PARITY(R5),(SP) : SET PARITY, STOP, CHARACTER SIZE

BICW #^XF007,(SP) : CLEAR SPECIAL FIELDS

BBS #UCBSV_TT_DSBL- : SKIP CLOCK ENABLE IF LINE DISABLED

UCBSB_TT_MAINT(R5),38

BISW #<CDZLPRSM_CLOCK>,(SP)

BISW UCBSW_UNIT(R5),(SP) : SET LINE NUMBER

BITW #DZCSRSM_MODE,(R4) : DZ32 CONTROLLER?

BNEQ 108 : YES

CVTLW (SP)+,2(R4) : INSERT AS LINE PARAMETER

RSB

HANDLE DZ-32 SPECIFIC FUNCTIONS

CMPB UCBSW_TT_SPEED(R5) - : TRANSMIT/RECEIVE THE SAME

UCBSW_TT_SPEED+1(R5)

BEQL 58 : YES, NO SPLIT SPEED

TSTB UCBSW_TT_SPEED+1(R5) : RECEIVE SPEED SPECIFIED?

BEQL 58 : NO, NO SPLIT SPEED

BISW #DZLPRSM_SPLIT,(SP) : SET SPLIT SPEED BIT

BRB 58 : COMPLETE SETUP

08AE 1399 .SBTTL INITIALIZE DZ-11 MODEM POLLING

08AE 1400

08AE 1401

08AE 1402 ++

08AE 1403 DZSSET_MODEM - INIT MODEM POLLING

08AE 1404

08AE 1405 FUNCTIONAL DESCRIPTION:

08AE 1406

08AE 1407 INIT DZ-11 MODEM TRANSITION POLLING IF NOT ALREADY ACTIVE. LINK CRB

08AE 1408 FOR CURRENT LINE INTO MODEM TRANSITION POLLING LIST

08AE 1409

08AE 1410 INPUTS:

08AE 1411

08AE 1412 R5 - UCB ADDRESS

08AE 1413

08AE 1414 OUTPUTS:

08AE 1415

08AE 1416 R0-R4 USED

08AE 1417 --

08AE 1418

08AE 1419 DZ11SET_MODEM:

54 24 A5	DD	08AE 1420 MOVL UCBSL_CRB(R5),R4 : ADDRESS CRB
00000000'EF	D5	08B2 1421 TSTL DZSL_DIALUP : DZ-11 POLLING ALREADY ACTIVE?
18 12	0888 1422 BNEQ 5\$: YES, SKIP STARTUP	
38 BB	088A 1423 PUSHR #^M<R3,R4,R5>	
55 00000004'EF	DE	088C 1424 MOVAL DZSTIMQUEENT,R5 : ADDRESS OF TIMER ENTRY
OB A5 06	90	08C3 1425 MOVB #IPLS_QUEUEAST,TQESB_RQTYPE(R5) : SET FORK IPL
000008D0'EF	9F	08C7 1426 PUSHAB 48 : RETURN ADDRESS
0021 31	08CD 1427 BRW 30S : QUEUE FORK	
38 BA	08D0 1428 4\$: POPR #^M<R3,R4,R5>	
	08D2 1429 5\$: MOVAL CRBSL_DZ_MODEM(R4),R3 : ADDRESS OF DZ CRB THREAD	
51 53 18 A4	DE	08D2 1430 MOVAL DZSL_DIALUP,R1 : ADDRESS OF DZ TIMER LIST HEAD
00000000'EF	DE	08D6 1431 MOVL R1,R2
52 51 DO	08DD 1432	
	08E0 1433 : LINK CRB INTO DZ-11 MODEM POLLER LIST IF NEEDED	
53 62 D1	08E0 1434	
08 13 08E3 1435	1436 10\$: CMPL (R2),R3 : IS CRB ON LIST	
52 62 DO 08E5 1437	1438 BEQL 20\$: YES, DONE	
F6 12 08E8 1439	1439 MOVL (R2),R2 : POINT TO NEXT CRB	
63 61 DO 08EA 1440	1440 BNEQ 10\$: LOOK FOR NEXT	
61 53 DO 08ED 1441	1441 MOVL (R1),(R3) : LINK CRB AT LIST HEAD	
	1442 MOVL R3,(R1)	
05 08F0 1443 20\$: RSB		
08F1 1444 30\$: JSB G^EXESFORK : FORK TO QUEUE TIMER ENTRY		
00000000'GF 16 08F1 1446	08F7 1447 DSBINT #IPLS_SYNCH	
OC AS FA45 CF 00000000'GF 9E 08FD 1448	0903 1448 MOVAB #^DZSTIMER,TQESL_FPC(R5) : ADDRESS OF TIMER SERVICE ROUTINE	
	090B 1449 MOVL G^TTYSGL_DELTA,TQESQ_DELTA(R5) : INTERVAL IS SYSGEN PARAMETER	
OB A5 05 00000000'GF 90 090B 1451	090F 1452 MOVB #TQESC_SSREPT,TQESB_RQTYPE(R5)	
50 00000000'GF CO 0916 1453	091D 1454 MOVR G^EXESGQ_SYSTIME,R0	
51 00 00 000000'GF 16 0920 1455	1455 ADDL G^TTYSGL_DELTA,R0	
	ADWC #0,R1	
	JSB G^EXESINSTMQ : INSERT INTO TIMER QUEUE	

DZDRIVER
V04-000

- Port Driver for DZ-11 support
INITIALIZE DZ-11 MODEM POLLING

L 14

16-SEP-1984 02:24:50 VAX/VMS Macro V04-00
5-SEP-1984 04:15:55 [TTDRV.R.SRC]DZDRIVER.MAR;1

Page 34
(1)

05 0926 1456 ENBINT
0929 1457 RSB
092A 1458

; RESTORE IPL

DZ
V04

092A 1460
00000000 1461 .PSECT \$SS117_DATA,QUAD
0000 1462 :
0000 1463 : DZ-11 TIMER DATA STRUCTURES FOR DETECTION OF DZ-11 MODEM TRANSITIONS
0000 1464 :
0000 1465 :
0000 1466 DZSL_DIALUP:
00000000 1467 .LONG 0 : LINKED LIST OF DZ-11 CRB'S
0004 1468 : USING MODEM CONTROL
0004 1469 DZ\$TIMQUENT:
0004 1470 : TIMER QUE ENTRY USED TO
0004 1471 : SAMPLE DZ-11 MODEM SIGNALS
0004 1472 : ON PERIODIC BASIS
00000038 1473 .ALIGN QUAD
0038 1474 .BLKB TQE\$C_LENGTH
0038 1475 STO_TQE TQE\$W_SIZE,WORD,TQE\$C_LENGTH,DZ\$TIMQUENT
0038 1476 STO_TQE TQE\$B_TYPE,BYTE,DYN\$C_TQE,DZ\$TIMQUENT
0038 1477 STO_TQE TQE\$B_RQTYPE,BYTE,TQE\$C_SSREPT,DZ\$TIMQUENT
0038 1478 :
0038 1479 DZSEND: : End of driver
0038 1480 :
0038 1481 .END

SSS	= 00000020	R	02	DZ11\$DS_SET	= 00000285	R	03
SSSSSS	= 00000038	R	04	DZ11\$INTINP	= 000003E9	RG	03
SSOP	= 00000002			DZ11\$INTOUT	= 000006E9	RG	03
ATS_UBA	= 00000001			DZ11\$MAINT	= 0000028C	R	03
BIT...	= 00000010			DZ11\$RESUME	= 000005E6	R	03
CLASS_DDT	= 0000000C			DZ11\$SET_MODEM	= 000008AE	R	03
CLASS_DS_TRAN	= 00000000			DZ11\$STARTIO	= 00000546	RG	03
CLASS_GETNXT	= 00000020			DZ11\$VEC	= 00000038	R	03
CLASS_POWERFAIL	= 00000004			DZ11\$VECEND	= 00000070	R	03
CLASS_PUTNXT	= 00000014			DZ11\$XOFF	= 0000059A	R	03
CLASS_READERROR	= 00000008			DZ11\$XON	= 0000059A	R	03
CLASS_SETUP_UCB	= 0000001D			DZ11_BURST	= 0000074F	RR	03
CRBSB_DZ_CARRIER	= 0000001E			DZ11_CHAR	= 00000769	R	03
CRBSB_DZ_DTR	= 0000001C			DZ11_OUT_EXIT	= 000006DC	R	03
CRBSB_DZ_RING	= 00000008			DZ11_OUT_LOOP	= 000006E9	R	03
CRBSB_TT_TYPE	= 00000018			DZ11_PREPRT	= 00000786	R	03
CRBSL_DZ_MODEM	= 00000024			DZ11_START_BURST	= 00000748	R	03
CRBSL_INTD	= 00000048			DZ11_STOP	= 00000779	R	03
CRBSL_INTD2	= 00000042			DZ32\$DS_SET	= 000002ED	R	03
DCS_TERM	= 0000000C			DZ32\$INTINP	= 00000477	RG	03
DDBSL_DDT	= 00040000			DZ32\$INTOUT	= 000007A3	RG	03
DEVSM_AVL	= 00000002			DZ32\$MAINT	= 0000025C	R	03
DEVSM_CCL	= 04000000			DZ32\$RESUME	= 00000676	R	03
DEVSM_IDV	= 00000200			DZ32\$STARTIO	= 0000056C	RG	03
DEVSM_NNM	= 08000000			DZ32\$VEC	= 00000070	R	03
DEVSM_ODV	= 00000001			DZ32\$VECEND	= 000000A8	R	03
DEVSM_REC	= 00000004			DZ32\$XOFF	= 00000645	RR	03
DEVSM_TRM	= 00000038	R	02	DZ32\$XON	= 00000645	R	03
DPTSC_LENGTH	= 00000004			DZ32_BURST	= 00000814	R	03
DPTSC_VERSION	= 00000038	R	02	DZ32_CHAR	= 0000082E	R	03
DPTSINITAB	= 00000004			DZ32_OUT_EXIT	= 00000796	R	03
DPTSM_NOUNLOAD	= 000000D3	R	02	DZ32_OUT_LOOP	= 000007A3	R	03
DPTSREINITAB	= 00000000	R	02	DZ32_PREPRT	= 00000854	R	03
DPTSTAB	= 0000001E			DZ32_START_BURST	= 0000080D	RR	03
DPTSW_VECTOR	= 00000042			DZ32_STOP	= 0000083F	R	03
DTS_DZ11	= 00000043			DZCSRSM_CLEAR	= 00000010		
DTS_DZ32	= 00000005			DZCSRSM_DS_CHG	= 00000800		
DYNSC_CRB	= 00000006			DZCSRSM_DS_ENAB	= 00000002		
DYNSC_DDB	= 0000001E			DZCSRSM_LINE	= 00000700		
DYNSC_DPT	= 00000049			DZCSRSM_MAINT	= 00000008		
DYNSC_ORB	= 0000000F			DZCSRSM_MASTENAB	= 00000020		
DYNSC_TOE	= 00000010			DZCSRSM_MODE	= 00000001		
DYNSC_UCB	= 000005CC	R	03	DZCSRSM_RCVINT	= 00000040		
DZ\$ABORT	= 0000014A	R	03	DZCSRSM_RCVRDY	= 00000080		
DZ\$CTRL_ERROR	= 00000000	RG	03	DZCSRSM SNDINT	= 000004000		
DZ\$DDT	= 00000000	RG	02	DZCSRSM SNDRDY	= 00000000		
DZ\$DPT	= 00000038	R	04	DZCSRSS_CLEAR	= 00000001		
DZ\$END	= 000000AD	RG	03	DZCSRSS_DS_CHG	= 00000001		
DZ\$INITIAL	= 0000014B	RG	03	DZCSRSS_DS_ENAB	= 00000001		
DZ\$INITLINE	= 00000000	R	04	DZCSRSS_LINE	= 00000003		
DZ\$L_DIALUP	= 000000AC	R	03	DZCSRSS_MAINT	= 00000001		
DZ\$NULL	= 00000864	RR	03	DZCSRSS_MASTENAB	= 00000001		
DZ\$SET_LINE	= 000005C4	R	03	DZCSRSS_MODE	= 00000001		
DZ\$STOP	= 00000346	RR	03	DZCSRSS_RCVINT	= 00000001		
DZ\$TIMER	= 00000004	RR	04	DZCSRSS_RCVRDY	= 00000001		
DZ\$TIMOULENT	= 00000257	R	03	DZCSRSS SNDINT	= 00000001		
DZ\$UNIT_ERROR				DZCSRSS SNDRDY	= 00000001		

DZCSR\$V_CLEAR	= 00000004	EXESGL_ABSTIM	***** X 03
DZCSR\$V_DS_CHG	= 00000008	EXESGL_TENUSEC	***** X 03
DZCSR\$V_DS_ENAB	= 00000001	EXESGL_UBDELAY	***** X 03
DZCSR\$V_LINE	= 00000008	EXESGL_SYSTIME	***** X 03
DZCSR\$V_MAINT	= 00000003	EXESINSTIMQ	***** X 03
DZCSR\$V_MASTENAB	= 00000005	FUNCTAB_LEN	= 00000000
DZCSR\$V_MODE	= 00000000	IDBSB_TT_ENABLE	= 0000000E
DZCSR\$V_RCVINT	= 00000006	IDBSL_UCBLST	= 00000018
DZCSR\$V_RCVRDY	= 00000007	IOSM_LINE_OFF	= 00000200
DZCSR\$V SNDINT	= 0000000E	IOSM_LINE_ON	= 00000800
DZCSR\$V SNDRDY	= 0000000F	IOSM_LOOP	= 00000080
DZLC\$1SM_ACK	= 00008000	IOSM_LOOP_EXT	= 00001000
DZLC\$1SS_ACK	= 00000001	IOSM_UNLOOP	= 00000100
DZLC\$1SV_ACK	= 0000000F	IOC\$MNTVER	***** X 03
DZLPR\$M_CLOCK	= 00001000	IOC\$RETURN	***** X 03
DZLPR\$M_LINE	= 00000007	IPLS_QUEUEAST	= 00000006
DZLPR\$M_ODD	= 00000080	IPLS_SYNCH	= 00000008
DZLPR\$M_PARITY	= 00000040	MODEMSC_DATASET	= 00000003
DZLPR\$M_SIZE	= 00000018	MODEMSC_INIT	= 00000000
DZLPR\$M_SPEED	= 00000F00	ORB\$B_FLAGS	= 00000008
DZLPR\$M_SPLIT	= 00002000	ORB\$L_OWNER	= 00000000
DZLPR\$M_STOP	= 00000020	ORB\$M_PROT_16	= 00000001
DZLPR\$S_CLOCK	= 00000001	ORB\$W_PROT	= 00000018
DZLPR\$S_LINE	= 00000003	PORT_ABORT	= 00000020
DZLPR\$S_ODD	= 00000001	PORT_DS_SET	= 0000000C
DZLPR\$S_PARITY	= 00000001	PORT_LENGTH	= 00000038
DZLPR\$S_SIZE	= 00000002	PORT_MAINT	= 00000030
DZLPR\$S_SPEED	= 00000004	PORT_RESUME	= 00000024
DZLPR\$S_SPLIT	= 00000001	PORT_SET_LINE	= 00000008
DZLPR\$S_STOP	= 00000001	PORT_SET_MODEM	= 00000028
DZLPR\$V_CLOCK	= 0000000C	PORT_STARTIO	= 00000000
DZLPR\$V_LINE	= 00000000	PORT_STOP	= 00000018
DZLPR\$V_ODD	= 00000007	PORT_VECTOR	= 00000038 R 03
DZLPR\$V_PARITY	= 00000006	PORT_XOFF	= 00000014
DZLPR\$V_SIZE	= 00000003	PORT_XON	= 00000010
DZLPR\$V_SPEED	= 00000008	PRS_IPL	= 00000012
DZLPR\$V_SPLIT	= 0000000D	SIZ..	= 00000001
DZLPR\$V_STOP	= 00000005	SS\$_NORMAL	= 00000001
DZRCV\$M_BUF	= 000000FF	TQE\$B_RQTYPE	= 0000000B
DZRCV\$M_FRAMER	= 00002000	TQE\$B_TYPE	= 0000000A
DZRCV\$M_LINE	= 00000700	TQE\$C_LENGTH	= 00000030
DZRCV\$M_OVERRUN	= 00004000	TQE\$C_SSREPT	= 00000005
DZRCV\$M_PARERR	= 00001000	TQE\$L_FPC	= 0000000C
DZRCV\$M_VALID	= 00008000	TQE\$Q_DELTA	= 00000020
DZRCV\$S_BUF	= 00000008	TQE\$W_SIZE	= 00000008
DZRCV\$S_FRAMER	= 00000001	TT\$M_DS_CTS	= 00000010
DZRCV\$S_LINE	= 00000003	TT\$M_DS_DSR	= 00000080
DZRCV\$S_OVERRUN	= 00000001	TT\$V_DS_CARRIER	= 00000005
DZRCV\$S_PARERR	= 00000001	TT\$V_DS_DTR	= 00000001
DZRCV\$S_VALID	= 00000001	TT\$V_DS_RING	= 00000006
DZRCV\$V_BUF	= 00000000	TT\$V_MODEM	= 00000015
DZRCV\$V_FRAMER	= 0000000D	TTS_UNKNOWN	= 00000000
DZRCV\$V_LINE	= 00000008	TTY\$GB_DEFSPEED	***** X 02
DZRCV\$V_OVERRUN	= 0000000E	TTY\$GB_PARITY	***** X 02
DZRCV\$V_PARERR	= 0000000C	TTY\$GB_RSPEED	***** X 02
DZRCV\$V_VALID	= 0000000F	TTY\$GL_DEFCHAR	***** X 02
EXESFORR	***** X 03	TTY\$GL_DEFCHAR2	***** X 02

TTY\$GL_DELTA
 TTY\$GL_DPT
 TTY\$GL_OWNUIC
 TTY\$GW_DEFBUF
 TTY\$GW_PROT
 TTYSM_TANK_BURST
 TTYSM_TANK_HOLD
 TTYSM_TANK_PREMPT
 TTYSM_TANK_STOP
 TTYSV_PC_NOTIME
 TTYSV_TANK_BURST
 UCB\$B_DEVCLASS
 UCB\$B_DEVTYPE
 UCB\$B_DIPL
 UCB\$B_FIPL
 UCB\$B_TT_DEPARI
 UCB\$B_TT_DETYPE
 UCB\$B_TT_DS_RCV
 UCB\$B_TT_DS_TX
 UCB\$B_TT_MAINT
 UCB\$B_TT_OUTTYPE
 UCB\$B_TT_PARITY
 UCB\$B_TT_PREMPT
 UCB\$C_TT_LENGTH
 UCB\$L_CRB
 UCB\$L_DDB
 UCB\$L_DDT
 UCB\$L_DEVCHAR
 UCB\$L_DEVCHAR2
 UCB\$L_DEVDEPEND
 UCB\$L_DEVDEPND2
 UCB\$L_DUETIM
 UCB\$L_TT_CLASS
 UCB\$L_TT_DECHA1
 UCB\$L_TT_DECHAR
 UCB\$L_TT_GETNXT
 UCB\$L_TT_OUTADR
 UCB\$L_TT_PORT
 UCB\$L_TT_PUTNXT
 UCB\$L_TT_RTIMOU
 UCB\$L_TT_WBLINK
 UCB\$L_TT_WFLINK
 UCB\$M_INT
 UCB\$M_ONLINE
 UCB\$M_TIM
 UCB\$M_TT_DSBL
 UCB\$V_INT
 UCB\$V_POWER
 UCB\$V_TT_DSBL
 UCBSW_DEVBUFSIZ
 UCBSW_STS
 UCBSW_TT_DESIZE
 UCBSW_TT_DESPEE
 UCBSW_TT_HOLD
 UCBSW_TT_OUTLEN
 UCBSW_TT_PRTCTL
 UCBSW_TT_SPEED

*****	X	03	UCBSW_TT_UNITBIT	= 00000106
*****	X	03	UCBSW_UNIT	= 00000054
*****	X	02	VEC\$L_IDB	= 00000008
*****	X	02	VEC\$L_INITIAL	= 0000000C
*****	X	02	VEC\$L_UNITINIT	= 00000018
= 00000800				
= 00000400				
= 00000100				
= 00000200				
= 00000000				
= 0000000B				
= 00000040				
= 00000041				
= 0000005E				
= 00000008				
= 000000EC				
= 000000F0				
= 00000124				
= 00000125				
= 0000012A				
= 0000010B				
= 00000F8				
= 0000010A				
= 00000134				
= 00000024				
= 00000028				
= 00000088				
= 00000038				
= 0000003C				
= 00000044				
= 00000048				
= 0000006C				
= 00000114				
= 000000C8				
= 000000C4				
= 0000010C				
= 0000011C				
= 00000118				
= 00000110				
= 000000B4				
= 000000D0				
= 000000CC				
= 00000002				
= 00000010				
= 00000001				
= 00000080				
= 00000001				
= 00000005				
= 00000007				
= 00000042				
= 00000064				
= 000000F1				
= 000000E8				
= 00000108				
= 00000120				
= 00000122				
= 000000F4				

```
+-----+
! Psect synopsis !
+-----+
```

PSECT name

```
-----  
: ABS .  
$ABSS  
$$$105_PROLOGUE  
$$$115_DRIVER  
$$$117_DATA
```

Allocation

	Allocation	PSECT No.	Attributes
00000000	(0.)	00 (0.)	NOPIC USR CON
00000000	(0.)	01 (1.)	NOPIC USR CON
000000DE	(222.)	02 (2.)	NOPIC USR CON
0000092A	(2346.)	03 (3.)	NOPIC USR CON
00000038	(56.)	04 (4.)	NOPIC USR CON

	ABS	REL	LCL NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE
CON	CON	CON	LCL NOSHR	EXE	RD	WRT	NOVEC	BYTE
CON	CON	CON	LCL NOSHR	EXE	RD	WRT	NOVEC	BYTE
CON	CON	CON	LCL NOSHR	EXE	RD	WRT	NOVEC	BYTE
CON	CON	CON	LCL NOSHR	EXE	RD	WRT	NOVEC	LONG
CON	CON	CON	LCL NOSHR	EXE	RD	WRT	NOVEC	QUAD

```
+-----+
! Performance indicators !
+-----+
```

Phase

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.07	00:00:00.37
Command processing	128	00:00:00.36	00:00:01.19
Pass 1	719	00:00:23.05	00:00:48.04
Symbol table sort	0	00:00:03.36	00:00:06.16
Pass 2	253	00:00:04.73	00:00:09.11
Symbol table output	35	00:00:00.20	00:00:00.26
Psect synopsis output	2	00:00:00.02	00:00:00.02
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	1169	00:00:31.80	00:01:05.15

The working set limit was 2250 pages.

187090 bytes (366 pages) of virtual memory were used to buffer the intermediate code.
 There were 170 pages of symbol table space allocated to hold 3041 non-local and 103 local symbols.
 1481 source lines were read in Pass 1, producing 25 object records in Pass 2.
 72 pages of virtual memory were used to define 67 macros.

```
+-----+
! Macro library statistics !
+-----+
```

Macro library name

```
$255$DUA28:[SYS.OBJ]LIB.MLB;1
-$255$DUA28:[SYSLIB]STARLET.MLB;2
TOTALS (all libraries)
```

Macros defined

```
33
12
45
```

3540 GETS were required to define 45 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:DZDRIVER/OBJ=OBJ\$:DZDRIVER MSRC\$:DZDRIVER/UPDATE=(ENH\$:DZDRIVER)+EXECMLS/LIB

0402 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

